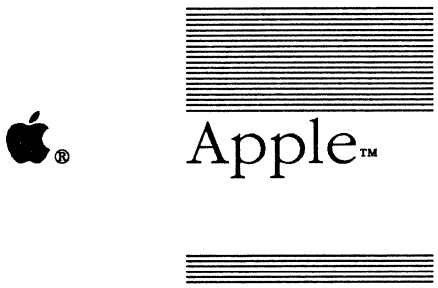


Apple™ A/UX Network System Administration

Final Draft

09/12/88

Sharon Everson

Developer Technical Publications

This document contains preliminary information.

It does not include:

- final technical information
- final editorial corrections
- Final art

© Apple Computer, Inc. 1988

APPLE COMPUTER, INC.

Copyright © 1988 by Apple Computer, Inc.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Computer, Inc. Printed in the United States of America.

Apple, the Apple logo, LaserWriter, and Macintosh are registered trademarks of Apple Computer, Inc.

ITC Avant Garde Gothic, ITC Garamond, and ITC Zapf Dingbats are registered trademarks of International Typeface Corporation.

Microsoft is a registered trademark of Microsoft Corporation.

POSTSCRIPT is a registered trademark of Adobe Systems Incorporated.

Varityper is a registered trademark, and VT600 is a trademark, of AM International, Inc.

Simultaneously published in the United States and Canada.

ISBN 0-201-xxxxx-x
ABCDEFGHIJ-DO-898
First printing, Nnnn 1988

2/21/88

A/UX Network System Administration

Contents

Preface

Chapter 1	Introduction
Chapter 2	Establishing a Two-System Network
Chapter 3	Initializing NFS
Chapter 4	Adding Yellow Pages Service
Chapter 5	Adding Systems to a Network
Chapter 6	Network Design Issues
Chapter 7	Network Management
Chapter 8	Tools for Checking System Status
Chapter 9	Troubleshooting
Chapter 10	Installing and Adminstering AppleTalk
Appendix A	Implementing a `sendmail` Facility
Appendix B	Name Server Operations Guide for BIND
Appendix C	Additional Reading

Preface

Conventions Used in This Manual

Throughout the A/UX manuals, words that must be typed exactly as shown or that would actually appear on the screen are in `Courier` type. Words that you must replace with actual values appear in *italics* (for example, *user-name* might have an actual value of `joe`). Key names appear in CAPS (for example, RETURN). Special terms are in **bold** type when they are introduced; many of these terms are also defined in the glossary in *A/UX System Overview*.

Syntax notation

All A/UX manuals use the following conventions to represent command syntax. A typical A/UX command has the form

`command` [*flag-option*] [*argument*]...

where:

`command` is a command name (the name of an executable file).

flag-option represents one or more flag options. Historically, flag options have the form

-[*opt*...]

where *opt* is a letter representing an option. The form of flag options varies from program to program. Note that with respect to flag options, the notation

`[-a][-b][-c]`

means you can select one or more letters from the list enclosed in brackets. If you select more than one letter, you use only one hyphen, for example, `-ab`.

argument represents an argument to the command, in this context usually a filename or symbols representing one or more filenames.

[]	surround an optional item.
...	follow an argument that may be repeated any number of times.
`Courier` type	anywhere in the syntax diagram indicates that characters must be typed literally as shown.
italics type	for an argument name indicates that a value must be supplied for that argument.

Other conventions used in this manual are:

<CR>	indicates that the RETURN key must be pressed.
^*x*	is an abbreviation for CONTROL-*x*, where *x* may be any key.
cmd(*sect*)	is a cross-reference to an A/UX reference manual; *cmd* is the name of a command, program, or other facility, and *sect* is the section number of the entry. For example, `netstat`(1N) refers to the `netstat` command, which is documented in section 1 of *A/UX Command Reference (M-Z and Games)*. You can also view the on-line reference page for the `netstat` command by entering `man netstat`

Chapter 1
Introduction

Contents

1. Overview of the A/UX network software 1
   1.1 Definition of terms 2

2. How to use this manual 4
   2.1 Starting a network 5
   2.2 Setting up network file service 5
   2.3 Setting up the Yellow Pages 5
   2.4 Adding systems to an existing network 5
   2.5 Setting up Internet forwarder systems and subnets 5
   2.6 Improving security on the network 5
   2.7 Providing network access to a new user 6
   2.8 Setting up a network mail system 6
   2.9 Performing backups of networked systems 6
   2.10 Modifying the Yellow Pages 6
   2.11 Finding information about systems 6
   2.12 Troubleshooting the network 6
   2.13 Printing to an AppleTalk printer 6
   2.14 Setting up or adding to a heterogeneous network environment 6
   2.15 Setting up an Internet domain server 7

Chapter 1

Introduction

1. Overview of the A/UX network software

This manual provides step-by-step instructions for setting up, maintaining, and troubleshooting A/UX™ network software. Here is a list of the network software included with A/UX:

- **B-NET** is Transmission Control Protocol/Internet Protocol (TCP/IP) based network software derived from the networking implementation developed at the University of California at Berkeley, and distributed in 4.3BSD. The B-NET software is part of the standard A/UX distribution; you enable it by using the scripts provided, which configure the B-NET software into the kernel. The B-NET software provides several TCP/IP/UDP-based utilities for the end-user.
- **NFS** (Network File System) is a facility developed and licensed by Sun Microsystems that allows you to export data to remote systems. NFS requires the B-NET software in order to run. Like the networking software, NFS is part of the standard A/UX distribution; you can configure it into the kernel by using the scripts provided. (The networking modules are included automatically when you configure NFS.) NFS provides transparent remote file access for end-users and their processes.
- The **Yellow Pages** (YP) are a distributed database service developed and licensed by Sun Microsystems that provides several key administrative files to Yellow Pages clients. The Yellow Pages require a B-NET kernel in order to run. The Yellow Pages are a network administration tool.
- **AppleTalk** is Apple's simple and flexible way to interconnect computers and peripheral devices. With A/UX, you can configure your system so that you can print to a LaserWriter on an AppleTalk network. You can set up your system for both an

AppleTalk network and a TCP/IP network.

A/UX also supports the following BSD networking features:

- **Subnets** are a BSD feature that allows a single Internet network number to support multiple networks within an organization (such as a school or company).
- **Internet domains** are part of the 4.3BSD BIND (Berkeley Internet Name Domain) distribution. The domain server (`named`) software serves Internet host names and addresses to domain clients (that is, systems using the `resolver` software). A/UX provides both the `named` and `resolver` code.
- **slip** (Serial Line Internet Protocol) is a program that lets you attach a serial line to the TCP/IP network. The `slip` program lets you use B-NET and NFS network programs such as `rlogin`, `rcp`, and `telnet` without requiring the Ethernet hardware. See `slip`(1N) and "Establishing the `slip` Environment" in Chapter 2 for more information.

1.1 Definition of terms

The following terms are used in this manual:

client Depending on the context, a system or a process that employs the resources provided to the network by a server.

directory hierarchy The collection of all files currently available to the system; that is, all files on currently mounted file systems.

export In NFS, to make the local file system(s) of a server system available to certain specified client systems.

file system A logical device that contains the data structures (directories, files, and inodes, among others) that implement all or part of the directory hierarchy.

host A machine on the network; often referred to as "local host" and "remote hosts."

Internet A group of networks interconnected by intelligent hosts referred to as **Internet forwarders** or **Internet routers.**

Internets can span several "nets." Therefore, the routing of data might involve several intermediate nodes on the way to the destination node.

Internet address
A 4-byte number that contains an Internet network number and a unique host number identifying each host on a network.

Internet broadcast address
An address understood by all hosts on a local network as a special address for broadcast packets.

Internet domain
A hierarchical database of hosts on the Internet.

Internet network number
A 1-, 2-, or 3-byte number that identifies a network. All hosts on that network use the same network number. An official Internet network number can be obtained from SRI International's Network Information Center. To connect with other networks, you need an official network number.

master server
In the Yellow Pages software, a designated host that contains the network's master database (including such files as `/etc/passwd`). Changes to this database are then made available to all systems using the Yellow Pages.

netmask A 32-bit string containing binary 1's and 0's. The 1's in the netmask define the "network part" of the Internet address, and the 0's in the mask define the "host part" of the Internet address.

protocol A well-known set of conventions (about how data is represented, checked, transmitted, and so forth) that must be implemented at both ends of a connection before any communication can take place.

server Depending on the context, a system or process that provides resources to the network.

slave server

A system that serves copies of the Yellow Pages databases obtained from the master server.

Yellow Pages domain

A concept used by the Yellow Pages to group hosts on the network. Using multiple domains can be a useful security or administrative measure in large network installations.

2. How to use this manual

This manual is divided into chapters that contain instructions for specific goals you may have in setting up, maintaining, or troubleshooting your network.

Following this introduction, Chapters 2, 3, and 4 are step-by-step guides to setting up a two-system network running B-NET software, NFS software, and the Yellow Pages. These guides instruct the administrator who is building a network from scratch using A/UX machines and serves as a guideline for the procedures involved in bringing up the network software on any system.

Chapter 5 describes how to add systems to a network. These instructions apply both when you, as the administrator, are building a larger network after setting up a working two-system network as described in Chapters 2, 3, and 4 and when you are adding an A/UX system to an existing BSD network.

Chapter 6 describes setting up an Internet forwarder system and subnets. This chapter provides a general overview of the Internet domain software when you are adding a system to a network that already supports a domain name server. Note that this chapter applies only to homogeneous networks of A/UX systems. If you are setting up a heterogeneous network, see Appendix C for a list of documentation that you should obtain from SRI International or the University of California at Berkeley.

Chapters 7, 8, and 9 deal with network management, security issues, tools for determining the status of networks and NFS systems, and solutions to specific error conditions on the network.

Chapter 10 describes setting up your system to print to a LaserWriter on an AppleTalk network.

2.1 Starting a network

If your goal is to install a network of A/UX systems, see Chapter 2, "Establishing a Two-System Network." This chapter describes how to bring A/UX systems up on a network running the BSD-derived networking software. This chapter also discusses how to set up a `slip` server. B-NET is a prerequisite to running NFS.

2.2 Setting up network file service

Chapter 3, "Initializing NFS," leads you through the procedures for setting up an NFS server and an NFS client machine on the two-system network created in Chapter 2.

2.3 Setting up the Yellow Pages

Chapter 4, "Adding Yellow Pages Service," discusses the Yellow Pages service and describes the process of creating the Yellow Pages databases and distributing them on the network.

2.4 Adding systems to an existing network

Chapter 5, "Adding Systems to a Network," describes various network design considerations, such as choosing machines to be servers or Internet forwarder systems, and explains how to establish a larger network from the small two-system network you established in the preceding chapters. This chapter also applies if you are adding a single system to a large existing network.

2.5 Setting up Internet forwarder systems and subnets

Chapter 6, "Network Design Issues," describes how to set up an Internet forwarder system between two discrete networks or between two subnets. Note that this chapter applies only to a homogeneous network of A/UX systems.

2.6 Improving security on the network

Chapter 7, "Network Management," describes how to increase security on networked or NFS systems.

2.7 Providing network access to a new user

Chapter 7, "Network Management," describes user administration using the Yellow Pages.

2.8 Setting up a network mail system

Chapter 7 describes how to make a minimal `sendmail` configuration file on a local network. Appendix A, "Implementing a `sendmail` Facility," contains Berkeley's installation and operation guide for `sendmail`.

2.9 Performing backups of networked systems

If you need a network backup system that uses one machine's backup medium, see "Backing Up Networked Systems" in Chapter 7.

2.10 Modifying the Yellow Pages

Chapter 7 also describes the tools used in modifying the Yellow Pages. See this chapter for instructions on moving the function of a Yellow Pages master server to another machine.

2.11 Finding information about systems

A system administrator often needs to determine the various parameters of the network, both when it is functioning normally and when it is malfunctioning. Chapter 8, "Tools for Checking System Status," is an introduction to the appropriate tools.

2.12 Troubleshooting the network

Chapter 9, "Troubleshooting," describes general error conditions for B-NET, NFS, and Yellow Pages, and provides recommended solutions. Information about specific error messages is also included.

2.13 Printing to an AppleTalk printer

Chapter 10 describes how to set up your system to print to a LaserWriter on an AppleTalk network.

2.14 Setting up or adding to a heterogeneous network environment

Appendix C, "Additional Reading," lists SRI's requests for comments (RFCs), materials from the University of California at Berkeley, and other required reading for those setting up subnets or discrete networks composed of different types of systems.

2.15 Setting up an Internet domain server

See Appendix B, "Name Server Operations Guide for BIND," for information on the Berkeley Internet Name Domain (BIND) from the University of California at Berkeley. Appendix C lists SRI's requests for comments (RFCs) and other required reading for those setting up Internet domain servers.

Chapter 2
Establishing a Two-System Network

Contents

1. Introduction 1
   1.1 Prerequisites to running B-NET 2

2. Preliminary steps 3
   2.1 A/UX network information files 4
   2.2 Choosing a host name 4
   2.3 Obtaining an Internet address 5
   2.4 Determining the Internet broadcast address 7
   2.5 Determining the netmask 8

3. Installing B-NET 8
   3.1 Modifying **/etc/inittab** 9
   3.2 Listing other network hosts 11
   3.3 Allowing open access (optional) 11
   3.4 Creating a B-NET or NFS kernel 12
   3.5 Rebooting with a new kernel 14
   3.6 Testing the network software 16

4. Bringing up another system on the network 17

5. Testing network communication 17
   5.1 Remote login 18

6. Establishing the **slip** environment 19
   6.1 Building a **slip** kernel 20
   6.2 Configuring the **/etc/slip.config** file . . . 20
   6.3 Configuring the **/etc/hosts** file 20
   6.4 Configuring the **/etc/slip.hosts** file 21
   6.5 Enabling **slip** on the server 21

7. Other required network system files 22

Figures

Figure 2-1. A two-system network 2

Tables

Table 2-1. A/UX network information files 4
Table 2-2. Internet addresses 6

Chapter 2

Establishing a Two-System Network

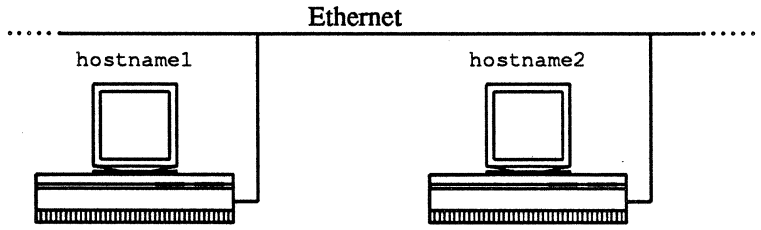
1. Introduction

This chapter presents step-by-step instructions for establishing a two-host TCP/IP network of A/UX machines. After you have completed these steps, the machines you have connected by Ethernet or serial line will be able to communicate. Although a complex network may be your goal, it is easier to make a small network and then expand it than to install the larger network at the beginning. After the two-system network is established, you can enlarge it to include printers, other CPUs, shared file systems, and so on. Instructions for expanding the network are given in later chapters.

The fundamental network that you will establish by following the steps in this chapter consists of two computers connected by a cable (see Figure 2-1), usually an Ethernet cable.

If you do not have Ethernet hardware, you can use the `slip`(1N) program to allow the serial lines on your A/UX machine to connect to the network. If you plan to use `slip`, you must build either a B-NET or NFS kernel. Follow the directions in this chapter to build a networking kernel: ignore the details specific to the Ethernet hardware. Then follow the instructions in "Establishing the `slip` Environment," later in this chapter.

Figure 2-1. A two-system network



Each computer runs software that allows communication between systems.

1.1 Prerequisites to running B-NET

The A/UX system includes all the software needed for both the TCP/IP network (B-NET) and the Network File System facility (NFS). The process of activating the software consists of supplying required information in system files, configuring system files to start programs automatically (**daemons**), running installation scripts, and rebooting with a new A/UX kernel.

For A/UX machines to communicate with each other across an Ethernet, the following hardware is required:

- an Ethernet card in each computer
- coaxial Ethernet cable, either the regular or the thin variety
- transceivers and transceiver cable (for regular Ethernet cable; not required for thin Ethernet)
- terminators

Instructions for Ethernet hardware installation are not included in this manual. See the documentation that accompanies your Ethernet hardware.

For A/UX machines to communicate with each other using `slip`, the following hardware is required:

- serial cable for direct connection
- or, a modem.

2. Preliminary steps

After installing the necessary network hardware, you need to obtain certain information for each machine. If you have installed an Ethernet card from a vendor other than Apple, refer to your vendor's documentation for instructions on the information you must supply.

If you are bringing up the A/UX systems for the first time (and an Apple EtherTalk card is installed), a startup script prompts you to enter the following information:

- system's host name
- Yellow Pages domain name
- Internet address
- Internet broadcast address
- netmask

See "Choosing a Host Name" for information about the system's host name and Yellow Pages domain name.

If you are establishing a `slip` connection, you are not prompted for the last three items. You need to edit certain files to contain this information, as described in "Establishing the `slip` Environment."

If the systems have already been running A/UX but you are putting them on the network for the first time, the system's host name and a domain name for the system should already be set. In this case, when a networking kernel is built and you have the EtherTalk hardware installed, the system prompts you for the

- Internet address
- Internet broadcast address
- netmask

The information you enter in response to these three prompts is stored in the A/UX network information files described in the next section.

See "Obtaining an Internet Address," "Determining the Internet Broadcast Address," and "Determining the Netmask" for more information about these prompts.

2.1 A/UX network information files

The A/UX standard distribution uses the files shown in Table 2-1 to store the required network information.

Table 2-1. A/UX network information files

Information	A/UX system file
host name	`/etc/HOSTNAME` (1st field)
domain name	`/etc/HOSTNAME` (2nd field)
Ethernet logical unit number	`/etc/NETADDRS` (1st field)
Internet address	`/etc/NETADDRS` (2nd field)
Internet broadcast address	`/etc/NETADDRS` (3rd field)
netmask	`/etc/NETADDRS` (4th field)

See "Creating a B-NET or NFS Kernel" for more information about the prompts displayed when you reboot the system.

Note that you can generate the prompts again to change the system's host name, Internet address, and so on, by removing the `/etc/HOSTNAME` or `/etc/NETADDRS` files and rebooting. Or, you can change the system's host name, Internet address, and so on, by simply editing these files with a text editor and rebooting.

2.2 Choosing a host name

Every network machine must have a name. If a system does not have a name, choose a name for it. The host name you choose must be no longer than 31 characters; should not include metacharacters such as `!`, `\`, `?`, or `*`; and must be unique on your network. The host name is public and used by everyone on the network to access files, write to users, and so forth.

The two machines set up on the network in this chapter are referred to as `hostname1` and `hostname2`.

Note: If you intend to use NFS and the Yellow Pages, you should also determine the right Yellow Pages domain name for

this machine. If you are not sure, you can enter a string (the same host name restrictions apply) and change the domain name later, either temporarily by using the `domainname` command, or permanently by editing the second field of `/etc/NETADDRS` and rebooting. However, do not use the `domainname` command to change your domain name if the Yellow Pages daemons are running on your machine. If this is your situation, first kill the Yellow Pages daemons, then use the `domainname` command and restart the Yellow Pages daemons if you want to temporarily change your domain name. See Chapter 4, "Adding Yellow Pages Service," for more information about domains.

2.3 Obtaining an Internet address

Machines are linked via networks, which themselves may be linked. For such a system to work, every network must have a unique Internet address. In a local area network that exists separately and never connects with another network, it doesn't matter if local network addresses duplicate addresses on another network. However, if such a local network were to connect with another network, a duplicate address would cause mass confusion in the network software. If you are sure your network will never connect with another network on the Internet, you can construct your own numbers as described in the note at the end of this section. However, if there is a possibility your site will want to communicate on the Internet, you need to obtain a unique network number and construct your Internet addresses from it.

An **Internet address** is a 4-byte number and is divided into class A, class B, and class C network numbers as shown in Table 2-2. The network number identifies the network itself, and the unique host number appended to it identifies each host on the network.

Table 2-2. Internet addresses

Network class	Number of bytes in network number	Range of numbers in first byte (in decimal)	Number of bytes in host number
A	1	1–126	3
B	2	128–191	2
C	3	192–223	1

Note that 0 and numbers over 223 are reserved for special purposes and should not be assigned to specific hosts on the network. The number 127 is reserved for the loopback driver.

Network numbers are assigned by a clearinghouse run by SRI International in Menlo Park, California. To obtain a unique network number for your site, call "Hostmaster" at

(800) 235-3155

You will be assigned a network number such as

```
128.8
```

Because the value of the first byte is within the range 128 to 191, this is a class B network number. You may be assigned a class A, class B, or class C network number. Unless there is more than one physical Ethernet at your site, or you are integrating a `slip` connection into an existing Ethernet, you should use the same network number for all hosts on the network. See "Routing and Forwarding" in Chapter 6 for information about using more than one distinct network number at a site.

After you have been assigned a network number, assign Internet addresses to machines on your network by appending a unique host number to your network number. If you are using a class B network number, the "network part" of the address is 2 bytes and the "host part" of the address is 2 bytes.

To avoid confusing the host address with a broadcast address, make sure that the bytes in the host field do not all contain 0's or 1's (for

example, do not assign `128.8.0.0` or `128.8.255.255` as an Internet address).

For example, if your network number is `128.8`, you can assign host numbers `1.1` and `1.2` to two hosts as follows:

```
128.8.1.1       hostname1
128.8.1.2       hostname2
```

These numbers indicate that the two machines are on the same network but are uniquely identified by the host number.

> *Note:* If time is short, you can use arbitrary "dummy" numbers until you receive your Internet Network number (but you should not connect your network to other networks). These dummy numbers should follow the conventions just described; that is, the network number must be the same for each machine on the network, and the host number must be unique for each machine on the network. A dummy class B network number should be of the form
>
> *X.Y.n.m*
>
> where *X* and *Y* are the network part of the address (and are the same for all hosts on the network) and *n* and *m* are host numbers that (together) are unique for each host on the network. *Do not connect your network(s) to another network without first obtaining an official network number.*

2.4 Determining the Internet broadcast address

The **Internet broadcast address** is understood by all hosts on a local network as a special address for broadcast packets. The Internet broadcast address is defined as the Internet address with a host part of all binary 1's. This is the broadcast method used by A/UX and by 4.3BSD.

The address used by 4.2BSD was the Internet address with a host part of 0. Although A/UX uses the standard broadcast address (all binary 1's) by default, A/UX allows you to set the broadcast address and netmask for each interface. You do so by responding to prompts when

you create an NFS or B-NET kernel.

For example, if the network number is

```
128.8
```

and all systems on the network use the standard broadcast address of all binary 1's, the broadcast address you should enter at the prompt is

```
128.8.255.255
```

since 255 decimal is equivalent to eight binary 1's (1 byte). (For the class C network number `192.11.9`, the broadcast address is `192.11.9.255`. For the class A network number `110`, the broadcast address is `110.255.255.255`.)

2.5 Determining the netmask

The **netmask** is a 32-bit string containing binary 1's and 0's. The 1's in the netmask define the network part of the Internet address, and the 0's in the mask define the host part of the Internet address. If you are setting up a simple two-system network of Macintosh® II computers, you may not need to redefine the network part of the systems' Internet address. For example, if you are using a class B network number, the netmask

```
0xffff0000
```

specifies that the first 2 bytes make up the network part of the address and that the last 2 bytes make up the host part. (That is, for a class B network number you do not redefine the network part of the address.)

See "Subnets" in Chapter 6 for more information about using netmasks in a homogeneous environment of Macintosh II computers. See the additional documentation listed in Appendix C under "Related RFCs" for information about subnets, netmasks, and broadcasts in general.

3. Installing B-NET

It is usually safer to bring up new programs in single-user mode than multi-user mode. Single-user mode is recommended for bringing a system up on the network. The following is a summary of the installation procedure. After installing the network hardware:

1. Bring the system up in single-user mode.
2. Edit `/etc/inittab` to start the network daemons.
3. Add information about other network hosts to `/etc/hosts` and (if desired) `/etc/hosts.equiv` and `$HOME/.rhosts`.
4. Create a B-NET, B-NET with `slip`, NFS, or NFS with `slip` kernel.
5. Reboot and enter `launch newunix` from the stand-alone shell (or run the `autoconfig` command to build a networking kernel and reboot), and answer the prompts that appear on your screen during reboot.

In this chapter the names of the first two machines you bring up on the network are `hostname1` and `hostname2`.

3.1 Modifying `/etc/inittab`

Once you have brought the system up in single-user mode:

1. Open `/etc/inittab` with a text editor. Change the status field (shown in bold) on the following lines from

   ```
   nfs0:2:off:/etc/portmap
   net4:2:off:/usr/etc/in.routed
   net5:2:off:/usr/etc/in.rwhod
   net9:2:off:/etc/inetd
   ```

   to

   ```
   nfs0:2:wait:/etc/portmap
   net4:2:wait:/usr/etc/in.routed
   net5:2:once:/usr/etc/in.rwhod
   net9:2:respawn:/etc/inetd
   ```

   and save the new version of `/etc/inittab`.

   You do not have to run all the network daemons listed in `/etc/inittab`. In particular, you may want to leave some of the following daemons off the first time you bring up your network:

- ☐ The routing daemon (`routed`)

  ```
  net4:2:wait:/usr/etc/in.routed
  ```

  manages network routing tables, but is not necessary on a simple one-cable network. See Chapter 6, "Network Design Issues," for information on multiple networks.

- ☐ The `rwho` daemon (`rwhod`)

  ```
  net5:2:once:/usr/etc/in.rwhod
  ```

  maintains the database used by the optional `rwho` and `ruptime` commands. The `rwho` daemon broadcasts frequently and may cause problems in a large or heterogeneous environment.

- ☐ The `/etc/portmap` daemon

  ```
  nfs0:2:wait:/etc/portmap
  ```

  is necessary if you intend to use remote procedure calls (RPCs), as with NFS. See the file `/etc/rpc` for a list of RPC services. For networking services, be sure to set the status field to `wait`.

2. Check `/etc/servers`. If a daemon is listed in this file, it is invoked by `inetd` from `/etc/inittab`. The contents of the file should look like this:

```
ftp     tcp    /usr/etc/in.ftpd
telnet  tcp    /usr/etc/in.telnetd
shell   tcp    /etc/in.remshd
login   tcp    /etc/in.rlogind
exec    tcp    /usr/etc/in.rexecd
tftp    udp    /usr/etc/in.tftpd
talk    udp    /usr/etc/in.talkd
finger  tcp    /usr/etc/in.fingerd
rpc     udp    /usr/etc/rpc.rstatd   100001   1-2
rpc     udp    /usr/etc/rpc.rwalld   100008   1
rpc     udp    /usr/etc/rpc.mountd   100005   1
rpc     udp    /usr/etc/rpc.rusersd  100002   1-2
rpc     udp    /usr/etc/rpc.sprayd   100012   1
comsat  udp    /etc/comsat
```

3.2 Listing other network hosts

You should add information about other network hosts to `/etc/hosts` on `hostname1`.

1. Open `/etc/hosts` with a text editor. This file already contains the loopback address (in hexadecimal):

   ```
   0x7F.0x00.0x00.0x01    loop  lo  loo localhost
   ```

2. Add an entry for at least one system that you will add to this network (in either hexadecimal or decimal numbers); for example,

   ```
   0x7F.0x00.0x00.0x01    loop  lo  loo localhost
   128.8.1.2              hostname2  h2
   ```

   See `hosts`(4) in *A/UX Programmer's Reference* for more information about this file. Note that you do not need to add an entry for the address(es) of the local host because this occurs automatically at reboot.

3.3 Allowing open access (optional)

Note: A hostname entry in `/etc/hosts.equiv` allows non-root users from the specified host (who also have an entry in the local `/etc/passwd`) to access the local host remotely

by using `rcp`, `remsh`, and `rlogin` without supplying a password. Note that for security reasons, a superuser on an equivalent host must supply a password unless the root account is set up to allow remote root account access.

To make the system more secure, you can use individual `$HOME/.rhosts` files instead of `/etc/hosts.equiv`. See `hosts.equiv` (4), `rlogin` (1N), and also "Network Security" in Chapter 7.

To allow all users (except root) on another host on the network to use `rcp`, `remsh`, or `rlogin` on `hostname1` without supplying a password:

1. Open `/etc/hosts.equiv` with a text editor. It should contain one line for the loopback driver:

   ```
   loop
   ```

2. Add a line for other hosts, for example,

   ```
   hostname2
   loop
   ```

3.4 Creating a B-NET or NFS kernel

To run B-NET or NFS software, you must create a kernel containing the appropriate software. A B-NET kernel will run only B-NET, whereas an NFS kernel will run both B-NET and NFS. The only advantage of making a B-NET kernel (assuming you never plan to run NFS) is one of memory size—a B-NET kernel uses less memory than an NFS kernel does.

If you want to run `slip`, you must create a B-NET or NFS kernel with support for `slip`. To use `slip`, follow the directions in this section and in "Rebooting with a New Kernel" until you are instructed to proceed to the section "Establishing a `slip` Environment."

To create a networking kernel:

1. Make sure that `root` can execute the required installation scripts by setting the permissions:

```
chmod 755 /etc/newunix
chmod 755 /etc/install.d/*
```

2. To create a B-NET kernel, enter

   ```
   /etc/newunix bnet
   ```

3. To create a B-NET kernel with support for `slip`, enter

   ```
   /etc/newunix bnet slip
   ```

4. To create an NFS kernel, enter

   ```
   /etc/newunix nfs
   ```

5. To create an NFS kernel with support for `slip`, enter

   ```
   /etc/newunix nfs slip
   ```

You can follow one of two equivalent procedures at this point. The two procedures produce the same result: a new B-NET or NFS kernel. *Use only one of these procedures.*

Procedure 1:

1. Enter

   ```
   sync; sync; reboot
   ```

2. Cancel the automatic boot procedure by clicking Cancel or by pressing the COMMAND key together with a period (`.`). This puts you in the Stand-Alone SHell (see `sash`(8) in *A/UX System Administrator's Reference*).

3. Enter

   ```
   launch newunix
   ```

   from the stand-alone shell, and A/UX will begin booting.

4. You will be prompted whether or not to check the root file system. Enter `y` for yes. (If this check causes your system to reboot, cancel the automatic boot procedure from the stand-alone shell, and then reenter the `launch newunix` command.)

   You will see a series of messages. Disregard the warnings. Your system will then automatically reboot. Let A/UX boot

again to single-user mode.

Procedure 2:

1. Enter

   ```
   /etc/autoconfig -v -o /unix -S /etc/startup
   ```

   You will see a series of messages. Disregard the warnings.

2. Enter

   ```
   sync; sync; reboot
   ```

3. As your system reboots, you will be prompted whether or not to check the root file system. Enter `y` for yes.

3.5 Rebooting with a new kernel

While the system is booting, you will be prompted for several kinds of information. The accompanying list shows the different prompts you may see when you boot your new kernel. Which prompts appear on your screen depend on the kernel you are booting and the prompts you may have answered during a previous boot.

Note: Use the DELETE key to fix typing mistakes on these lines before pressing RETURN. Do not use the BACKSPACE key for this purpose. See `ttytype`(5) in *A/UX Programmer's Reference* for more information.

```
Please enter a hostname (it must be unique):
```

Type the system's host name and press RETURN. See "Choosing a Host Name," earlier in this chapter, for restrictions and naming conventions.

```
Please enter a domainname:
```

See "Yellow Pages Domains" in Chapter 4 for an explanation of domains. If you do not intend to use the Yellow Pages, enter any string that is no longer than 31 characters and does not include metacharacters such as `!`, `\`, `?`, `*`, or RETURN. Host and domain names are stored in the `/etc/HOSTNAME` file, which is read by the `hostname` command when

`/etc/startup.d/ae6` is called by `/etc/startup`. The `/etc/startup` script is run by `/etc/sysinitrc` at boot time.

If you have created a B-NET or NFS kernel with `slip` and you don't have the Ethernet hardware, you won't see any of the prompts shown in the rest of this section. In this case, skip to the section "Establishing a `slip` Environment." Then to add another system to your network, follow the directions in "Bringing up Another System on the Network."

As the system is booting, it notices the new Ethernet card and prompts you for the Internet address, broadcast address, and netmask associated with the new card:

```
1 Ethernet card(s) installed

ae0:  Please enter an Internet address:
```

Type the system's Internet address and press RETURN. For this example system, you would enter

```
128.8.1.1
```

```
ae0:  Please enter an Internet Broadcast address:
```

Type the system's Internet broadcast address and press RETURN. For example, using the example network set up in this chapter, you would enter

```
128.8.255.255
```

See "Determining the Internet Broadcast Address," earlier in this chapter, for more information.

```
ae0:  Please enter a netmask [none]:
```

A netmask is necessary only if your network uses subnet routing. If you are not setting up subnet routing, press RETURN to accept "none" as the default response. Otherwise, type the netmask you are using and press RETURN. For example, using the example network set up in this chapter and the broadcast address set above, you would enter

```
0xffff0000
```

Note that you must type the `0x` (hexadecimal) prefix and all eight digits of the netmask. Which number you enter here depends on which broadcast method you are using and on how the network will be configured. See "Determining the Netmask" earlier in this chapter for more information.

The `/etc/start.d/ae6` script stores the Internet address, Internet broadcast address, and netmask in `/etc/NETADDRS`. It also appends an entry with the host name and Internet address to the end of `/etc/hosts` with a date stamp in the comment field.

Note: If you forced the host name and domain name prompt to appear by removing the `/etc/NETADDRS` file, you may now have duplicate entries for this host *at the same Internet address* in `/etc/hosts`. In most cases, these duplicate entries cause no harm. However, if this host is a Yellow Pages master server, be sure to remove these duplicate entries from `/etc/hosts`. Otherwise, inconsistent Yellow Pages behavior may result.

After you have entered the appropriate information, the boot procedure completes as usual. Given the sample information shown in this section, A/UX prints the lines:

```
ae0: hostname is hostname1 with IP address of 128.8.1.1
     broadcasting with 128.8.255.255
ae0: 128.8.1.1 netmask ffff0000 flags=63<UP,BROADCAST,
     NOTRAILERS,RUNNING> broadcast: 128.8.255.255
INIT: SINGLE USER MODE
```

3.6 Testing the network software

The simplest test of the network software is to see whether the local system can talk to itself by using the loopback interface. To run this test:

1. Bring the system up in multi-user mode, and log in as `root`.
2. Enter

```
telnet loop
```

This command requests access to a remote login (through the network looped back to itself) on the local system. After a short time, you should see the login prompt:

```
A/UX Apple Computer, Inc. (hostname1)

login:
```

If this test fails, see Chapter 8, "Tools for Checking System Status," for information on utilities for checking the network.

4. Bringing up another system on the network

To put `hostname2` on the network, follow the same procedure you just followed for `hostname1`:

1. Bring the system up in single-user mode.
2. Edit `/etc/inittab` to start the network daemons (as previously described in "Modifying `/etc/inittab`").
3. Add information about other network hosts to `/etc/hosts` and (if desired) `/etc/hosts.equiv` and `$HOME/.rhosts` (as previously described in "Listing Other Network Hosts" and "Allowing Open Access").
4. Create a B-NET or NFS kernel (as previously described in "Creating a B-NET or NFS Kernel").
5. Reboot and enter `launch newunix` from the stand-alone shell (or run the `autoconfig` command to build a networking kernel and reboot), and answer the prompts that appear on your screen during reboot (as previously described in "Creating a B-NET or NFS Kernel").

5. Testing network communication

When you have rebooted `hostname2` and tested the network software with the loopback interface, you can use the `ping` command to determine if the network is working. From `hostname2` enter

```
/usr/etc/ping hostname1
```

If the network and both hosts are functional, `ping` produces a display similar to this one:

```
64 bytes from 128.8.1.1: icmp_seq=0. time=16. ms
64 bytes from 128.8.1.1: icmp_seq=1. time=16. ms
64 bytes from 128.8.1.1: icmp_seq=2. time=16. ms
64 bytes from 128.8.1.1: icmp_seq=3. time=16. ms

(interrupt)

----hostname1 PING Statistics----
4 packets transmitted, 4 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 16/16/16
```

Use the interrupt character (usually CONTROL-C) to stop the output of `ping` once the packets are being returned. The `ping` command prints statistics and exits.

This command works in either direction. You can also enter

```
/usr/etc/ping hostname2
```

from `hostname1`. The resulting display is similar to this:

```
64 bytes from 128.8.1.2: icmp_seq=0. time=16. ms
64 bytes from 128.8.1.2: icmp_seq=1. time=16. ms

(interrupt)

----hostname2 PING Statistics----
2 packets transmitted, 2 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 16/16/16
```

5.1 Remote login

If you have a login account on both hosts, you can log in to one from the other. For example, if you are currently logged in to `hostname1`, use the following `telnet` command to connect to `hostname2`:

```
telnet hostname2
```

If you are prompted for your login and password on the remote system, the test worked. Just press CONTROL-d at the remote login prompt. If you see the message

```
Trying...
```

followed by

```
Connection timed out.
```

then the test failed. See Chapters 8 and 9 for more information.

6. Establishing the `slip` environment

The `slip` program allows serial lines to interface to the TCP/IP network. The `slip` program lets you use a serial line to connect to remote machines without requiring Ethernet hardware. You can set up your machine as either a `slip` client or a `slip` server.

If you want to allow other users to attach their serial lines to the network, for instance, by using modems to dial your machine, then set up your machine as a `slip` server. After establishing a `slip` connection with your machine, dial-in users can use B-NET or NFS network programs to connect to your machine and to other machines connected to yours by the network. Note that both the `slip` server and `slip` client must run the router `/etc/in.routed` (set up in `/etc/inittab`) to access other hosts transparently.

If you want to access other machines using `slip`, then follow the directions in *A/UX Communication User's Guide* to set up your machine as a `slip` client.

The basic steps to configure your machine as a `slip` server are listed here. The subsections that follow explain the steps in more detail.

1. Build a networking kernel that includes support for `slip`.
2. Modify the `/etc/slip.config` file to contain a host address for each `slip` interface supported by that server.
3. Modify the `/etc/hosts` file to contain the Internet address of the `slip` client and `slip` server.

4. Modify the `/etc/slip.hosts` file to contain the Internet address and user name of each `slip` client.
5. Run `mkslipuser`.

6.1 Building a `slip` kernel

To set up your A/UX machine as a `slip` server, you must first create a new kernel. See "Creating a B-NET or NFS kernel," earlier in this chapter, for instructions.

After creating the networking kernel, you must modify the `/etc/slip.hosts`, `/etc/hosts`, and `/etc/slip.config` files. Then run `mkslipuser` to allow dial-in users to establish a `slip` connection to your machine.

6.2 Configuring the `/etc/slip.config` file

The `/etc/slip.config` file must be configured on the `slip` server to establish `slip` connections between the `slip` server and the `slip` client. The `/etc/slip.config` file must contain a host address for each `slip` interface supported by that server. Host addresses are listed on separate lines, and each line configures a separate serial interface.

Here is a sample `/etc/slip.config` file:

```
# slip.config configuration file
# Each line configures a serial line
#
128.120.254.3
128.120.254.3
```

A Macintosh II has two built-in serial interfaces. To use them both for `slip`, you must list host addresses as two separate lines in `/etc/slip.config`. In the example, the server has two `slip` interfaces, each using the same host address, available for user dial-up use.

6.3 Configuring the `/etc/hosts` file

You should modify the `/etc/hosts` file on the server so that you can use host names instead of network addresses in your network commands. The `/etc/hosts` file on the server should contain the

Internet address of the slip client. The Internet address of the slip server should also be included in this file. Here is a sample /etc/hosts file:

```
0x7F.0x00.0x00.0x01 loop lo loo localhost
128.120.254.3 hostname1 #slip server
128.120.253.1 hostname2 #slip client
```

The first line contains the loopback address; this line is always present in the /etc/hosts file. The second line is the Internet address and host name of the slip server. The third line is the client Internet address and host name used when the client makes a slip connection.

6.4 Configuring the /etc/slip.hosts file

The /etc/slip.hosts file is used to map user names on slip client machines to the Internet addresses of the slip clients. The /etc/slip.hosts file contains the Internet address used by each slip client when that user makes a slip connection to the slip server. Here is a sample /etc/slip.hosts file:

```
# dialup slip.hosts table
# maps usercodes to host addresses
#
128.120.253.1 joe
128.120.253.2 wendy
128.120.253.3 mike
128.120.253.4 linda
```

The Internet address in the first field is used when the user specified in the second field invokes slip.

6.5 Enabling slip on the server

After modifying the /etc/hosts, /etc/slip.hosts, and /etc/slip.config files, run

```
mkslipuser
```

to create the /etc/slip.user file. The slip.user file is not human readable. You can use the command dslipuser (1N) to display the contents of the user file, and to report the number of slip users on the system and the number of available slip interfaces.

Your system is now configured as a `slip` server.

To allow `slip` clients to use host names instead of network names when they establish a `slip` connection to your machine, instruct the users on `slip` client machines to modify their local `/etc/hosts` files. The `/etc/hosts` file on the `slip` client should contain the Internet address and host name used by the `slip` client when using `slip`. The Internet address and host name of the `slip` server should also be in this file.

7. Other required network system files

The B-NET software also requires the following files to be present:

`/etc/networks`

The networks available to the system should be listed here. For each network, there should be a single line with the following information:

network-name network-number aliases

Here is a sample `/etc/networks` file :

```
loopback  127
# Internet networks
#
arpanet   10   arpa
ucb-ether 46   ucbether
```

If you connect to an outside network, this file is normally created from the official network database maintained at the Network Information Center (NIC). You can also assign a name to the network(s) at your own site. Use a text editor to open `/etc/networks`, and add a line for your own network; for example,

```
ether-net 128.8.1  localnet ournet
```

Network names may contain any printable character other than a field delimiter (blank or tab), a new line, or a comment character (#).

`/etc/services`
: Information about known services available on the network is located here. These services are called by various network commands. This file seldom needs modification.

`/etc/protocols`
: Information about system protocols used by the network is located here. This file is modified only if the network is joining another network that uses protocols not listed in this file.

`/etc/ftpusers`
: Although this file is not required by the network software, the A/UX standard distribution contains a `root` entry in this file. This prevents remote `ftp` users from logging in as `root` on the local system.

Chapter 3
Initializing NFS

Contents

1. Introduction 1
   1.1 Overview of NFS 1

2. Installing NFS on a server machine 4
   2.1 Enabling the NFS daemons 4
   2.2 Adding lines to `/etc/exports` 5
   2.3 Creating an NFS kernel 6
   2.4 Checking that NFS service is running 6

3. Installing NFS on a client machine 7
   3.1 Checking `/etc/passwd` for a `nobody` entry 8
   3.2 Enabling the NFS daemons and `mount` command 8
   3.3 Creating an NFS kernel 9
   3.4 Checking that NFS service is running 9
   3.5 Testing NFS with a temporary remote mount 10
   3.6 Modifying `/etc/fstab` 12
   3.7 Mounting a file system specified in `/etc/fstab` 15

Figures

Figure 3-1. Remotely mounted manual pages 3

Tables

Table 3-1. NFS mount options 13

Chapter 3

Initializing NFS

1. Introduction

Note: Chapter 2 described how to install a two-system B-NET network. A functioning network is a prerequisite to running NFS. See "Creating a B-NET or NFS Kernel" in Chapter 2, where there is an option to build the NFS kernel while bringing the machine up on the network. If you chose to build the NFS kernel at that time, you do not need to do it again. In this case, simply skip "Creating an NFS Kernel" in this chapter.

The first part of this chapter describes how to set up a machine as an NFS file **server** to allow access to its files. The second part of this chapter explains how to set up a machine as an NFS **client** to remotely mount a server's files. After completing these two sections, you will have one NFS server and one client. If your goal is to configure a large local area network, you can set up additional servers and clients by repeating the steps described in this chapter.

1.1 Overview of NFS

A **file system** is a logical device that contains the data structures (directories, files, and inodes, among others) that implement all or part of the directory hierarchy. The **directory hierarchy** is the collection of all files currently available to the system, that is, all files on currently mounted file systems (see `mount`(1M) in *A/UX System Administrator's Reference*). The general term **hierarchy** is used for the collection of files and subdirectories of a given directory structure; for example, the `/usr` hierarchy means all files and directories under the `/usr` directory.

Note: The current A/UX distribution for a Macintosh II on an 80-MB (megabyte) hard disk contains only one user-accessible

partition: the root file system partition. Thus, to export a particular directory hierarchy of the root file system, you must first export the entire root file system; then you may export any directory hierarchy you wish. See "Testing NFS with a Temporary Remote Mount" in this chapter for more information.

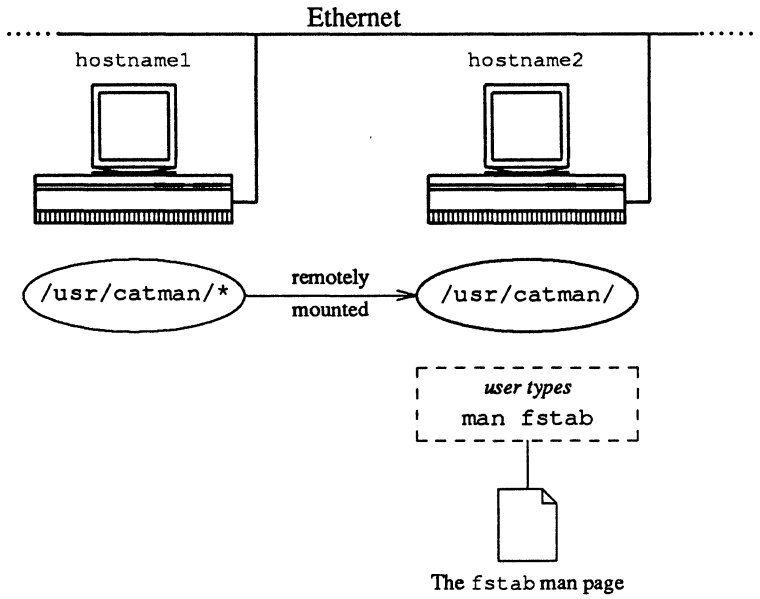
In NFS, a server machine explicitly **exports** (allows remote access to) its file systems. A client machine can remotely mount the exported file system or a hierarchy of that file system by specifying that hierarchy in `/etc/fstab`. When a client system has remotely mounted data from a server, users and processes on the client system can transparently access that data as if it were stored locally.

For example, in Figure 3-1 `hostname1` is a Macintosh II exporting its root file system to `hostname2`. The machine `hostname2` has remotely mounted the `/usr/catman` hierarchy from `hostname1` on a mount-point directory named `/usr/catman`. When a user on `hostname2` enters

```
man fstab
```

the `man` command on `hostname2` accesses the on-line manual page as if it were on the local disk, and the `fstab` manual page is displayed on the user's screen.

Figure 3-1. Remotely mounted manual pages



The NFS service works as follows: A client's `mount` request talks to the server's `mountd` daemon. The daemon checks the client's access permission and, if correct, returns a pointer to the requested file system. After the mount completes, access to that mount point and below goes through the pointer to the server's `nfsd`(1M) daemon by using a remote procedure call (RPC). Client kernel file access requests (delayed-write and read-ahead) are handled by the `biod`(1M) daemons on the client.

Thus, a process on a client machine can directly access files located on the server machine when the following conditions are met:

- The server has exported the file system in which the file is located.
- The client machine has mounted the remote file system on one of its hierarchies.

- The process is owned by a user who has permission to access the file.

2. Installing NFS on a server machine

A large disk or multiple disks on a system are probably the most important consideration in choosing a system to be an NFS server. A second criterion is the overall reliability of the server, especially when file systems are mounted with the "hard" option. The "hard" option ensures that a process accessing remote data will not complete until the read is successful. If the remote server is down, the process will hang until the remote server returns. See Table 3-1 for more information.

It is usually safer to bring up new programs in single-user mode and this is recommended for bringing up NFS. In this chapter, the NFS server is called `hostname1`. The basic steps for making `hostname1` an NFS server are listed below. The subsections that follow explain the steps in more detail.

1. Bring the system up in single-user mode. (If you are already up in multi-user mode, use the `shutdown` program to bring the system to single-user mode.)
2. Edit `/etc/inittab` to start the NFS daemons.
3. Edit `/etc/exports` to specify hosts eligible to mount file systems from this server.
4. If you have not yet built an NFS kernel, follow the procedure in "Creating a B-NET or NFS Kernel" in Chapter 2. If you have already built the NFS kernel in Chapter 2, put your system in multi-user mode.

2.1 Enabling the NFS daemons

Once you have brought the system up in single-user mode,

1. Open `/etc/inittab` with a text editor. Change the status field (shown in bold) in the following line from

   ```
   nfs3:2:off:/etc/nfsd 4
   ```

   to

```
nfs3:2:wait:/etc/nfsd 4
```

and save the new version of `/etc/inittab`.

2. The `mount` daemon, `rpc.mountd`, must be enabled for NFS to work. The `/etc/servers` file is listed below, with the required `rpc.mountd` line printed in bold:

```
ftp     tcp    /usr/etc/in.ftpd
telnet  tcp    /usr/etc/in.telnetd
shell   tcp    /etc/in.remshd
login   tcp    /etc/in.rlogind
exec    tcp    /usr/etc/in.rexecd
tftp    udp    /usr/etc/in.tftpd
talk    udp    /usr/etc/in.talkd
rpc     udp    /usr/etc/rpc.rstatd   100001   1-2
rpc     udp    /usr/etc/rpc.rwalld   100008   1
rpc     udp    /usr/etc/rpc.mountd   100005   1
```

Be sure that the line printed in bold is listed in the `/etc/servers` file, in which case the `inetd` daemon invokes `rpc.mountd`.

2.2 Adding lines to `/etc/exports`

NFS file servers use the `/etc/exports` file to control which file systems can be mounted by which systems on the network. Entries in `/etc/exports` specify a file system name that can be remotely mounted; the file system name is left-justified and may be followed by a list of host names or netgroup names separated by space or tab characters.

If a host name or netgroup name follows the file system name, export permissions are limited to the host(s) or netgroup(s) specified; otherwise, the file system is open to everyone. A number sign (#) anywhere on a line begins a comment to the end of the line. See "Creating a Netgroup File (Optional)" in Chapter 4 for more information about establishing network-wide groups.

Note that on an A/UX NFS server with a single disk, the entire root file system must be exported. For example,

```
/              # export to everyone
```

or

```
/  hostname2  # export to hostname2
```

After you have exported the root file system, you may want to list specific hierarchies that are resident on the local disk, such as `/usr/catman`. For example,

```
/              # export to everyone
/usr/catman    # export to everyone
```

Note that the specific reference to `/usr/catman` in `/etc/exports` on an A/UX server is simply an optional convenience to other systems. Listing a hierarchy in `/etc/exports` makes no difference in terms of export permissions once you have exported the root file system in which it is located.

To restrict access to the file system to `hostname2`, edit `/etc/exports` to read

```
/           hostname2 # export to hostname2
/usr/catman hostname2 # export to hostname2
```

To give open permissions, omit the host name.

2.3 Creating an NFS kernel

If you have previously made an NFS kernel, put your system in multi-user mode with the command

```
init 2
```

Otherwise, make an NFS kernel by following the procedure in "Creating a B-NET or NFS Kernel" in Chapter 2, reboot, and put the machine in multi-user mode.

2.4 Checking that NFS service is running

After the system has come up in multi-user mode,

1. Check that the appropriate daemons are running by using the command

   ```
   ps -ef | grep nfsd
   ```

   The response should be

```
root   81   1   0 01:27:17 ?     0:02 /etc/nfsd 4
root   82   1   0 01:27:17 ?     0:02 /etc/nfsd 4
root   83   1   0 01:27:17 ?     0:02 /etc/nfsd 4
root   84   1   0 01:27:17 ?     0:02 /etc/nfsd 4
```

(The numbers in each column may be different on your system). If the `nfsd` processes are not running, reboot and check again.

2. Check that a file system has been exported from this system by using the command

   ```
   showmount -e
   ```

   With the example file system exports, the response would be

   ```
   export list for hostname1:
   /              hostname2
   ```

3. Installing NFS on a client machine

In this chapter the NFS client is called `hostname2`. The basic steps for making `hostname2` an NFS client are listed below. The subsections that follow explain the steps in more detail.

1. Bring `hostname2` up in single-user mode (if you are already up in multi-user mode, use the `shutdown` program to bring the system to single-user mode), and make sure the network is running by attempting a `telnet` remote login to `hostname1`.
2. Check that `/etc/passwd` has a `nobody` entry.
3. Edit `/etc/inittab` to enable the NFS daemons and enable the automatic `mount` command.
4. If you have previously made an NFS kernel, put `hostname2` in multi-user mode with the command

   ```
   init 2
   ```

   Otherwise, make an NFS kernel by following the procedure in "Creating a B-NET or NFS kernel" in Chapter 2, reboot, and put the machine in multi-user mode.

5. Mount the remote file system.
6. Edit `/etc/fstab` to mount remote file systems whenever the system is rebooted.

3.1 Checking `/etc/passwd` for a `nobody` entry

For system security reasons, the superuser does not have access permissions on remote files. This is implemented by mapping UID 0 (`root`) to UID 65534 (unsigned representation of –2 in 2's complement notation) on all client machines. When UID 0 is mapped to UID 65534, the superuser on a client machine has the same permissions on remote files as a user with UID –2. Unless the files contained in remote file system's permissions allow access to "others," the superuser will not be allowed to look at them. See "Network Security" in Chapter 7 for more information.

All NFS client machines (including server machines that will also mount remote file systems) should have a `passwd` entry for `nobody` with UID 65534. Type

```
fgrep nobody /etc/passwd
```

The response should be

```
nobody:xxxxxxxxxxxxx:65534:65534:NFS generic user:
/tmp:/bin/noshell
```

(on one line).

If this line does not appear on the screen, modify `/etc/passwd` to include it.

3.2 Enabling the NFS daemons and `mount` command

The NFS daemons are invoked in `/etc/inittab`. There is also a `mount` command line in `/etc/inittab`; if this line is enabled, the entries in `/etc/fstab` will be remotely mounted on the client system whenever it reboots.

1. Open `/etc/inittab` with a text editor, and change the status field (shown in bold) in the following lines from

```
nfs4:2:off:/etc/biod 4
nfs8:2:off:/etc/mount -at nfs >/dev/syscon 2>&1
```

to

```
nfs4:2:wait:/etc/biod 4
nfs8:2:once:/etc/mount -at nfs >/dev/syscon 2>&1
```

Save the new version of `/etc/inittab`.

2. Check that the `/etc/servers` file contains the line

```
rpc     udp     /usr/etc/rpc.mountd    100005    1
```

3.3 Creating an NFS kernel

If you have previously made an NFS kernel, bring your system to the multi-user state with the command

```
init 2
```

Otherwise, make an NFS kernel by following the procedure in "Creating a B-NET or NFS Kernel" in Chapter 2, reboot, and put `hostname2` in multi-user mode.

3.4 Checking that NFS service is running

After the system has come up in multi-user mode:

1. Check that the appropriate daemons are running by typing

```
ps -ef | grep biod
```

The response should be

```
root  81  1  0 01:27:17 ?     0:02 /etc/biod 4
root  82  1  0 01:27:17 ?     0:02 /etc/biod 4
root  83  1  0 01:27:17 ?     0:02 /etc/biod 4
root  84  1  0 01:27:17 ?     0:02 /etc/biod 4
```

(The numbers in each column may be different on your system.) If the `biod` processes are not running, reboot and check again.

2. Check that the root file system is correctly exported from the server. On `hostname2` (the client), enter the command

```
showmount -e hostname1
```

This prints the file systems exported from `hostname1`. The example `/etc/exports` file in this chapter prints

```
export list for hostname1:
/               hostname2
```

3.5 Testing NFS with a temporary remote mount

The on-line manual pages take several megabytes of storage space. It is a useful space saver to import this hierarchy from one system to several machines.

Note: Usually a local mount-point must be created with open access permissions. It should be an empty directory; otherwise, its files become obscured and thereby made inaccessible by the file system mounted over it. However, it is often the case that a directory named `/usr/catman` already exists on A/UX. If this is the case, you do not need to create a new mount-point.

To remotely mount the `/usr/catman` hierarchy from `hostname1`:

1. Check that the local `/usr/catman` mount-point exists and is empty. If it is not empty, back it up and then remove its subdirectories:

   ```
   /bin/rm -r /usr/catman/*
   ```

2. Check permissions on the remote `/usr/catman` hierarchy. Type

   ```
   remsh hostname1 ls -ld /usr/catman
   ```

   You should see something like this:

   ```
   drwxr-xr-x 4 bin  bin   15 Mar 31  09:33 /usr/catman/
   ```

3. Check permissions on the local `/usr/catman` mount-point. Just as in local mounts, the mount-point directory must have access permissions that are at least as open as the remote file system being mounted. In addition, users' UIDs and GIDs must agree across systems. (See Chapter 4 for information on how using the Yellow Pages can facilitate these NFS permission

requirements.) In the case of `/usr/catman`, the local `man` command requires that the owner and group of the local mount-point directory match the remote hierarchy. Type

```
ls -ld /usr/catman
```

You should see something like this:

```
drwxr-xr-x 4 bin  bin   64 Sep 13 05:06 /usr/catman/
```

4. If the permissions check out, try a `mount` command. Type

   ```
   mount -o soft, ro hostname1:/usr/catman /usr/catman
   ```

   This command mounts the remote file system with the `soft` and `ro` (read-only) options.

   > *Note:* If the permissions of the topmost directory of the file system you are remotely mounting disallow write permissions for all users, you must mount the hierarchy with the `ro` (read-only) option. (Remember that write permission for `root` is not preserved across the network.) This procedure prevents errors when file access times are updated across the network. These errors would occur whether or not an explicit write was attempted on the hierarchy. See Table 3-1 for more information about `mount` options.

5. Check that the remote file system is mounted where you expected with the command

   ```
   mount
   ```

   This command displays the currently mounted file systems:

   ```
   /dev/dsk/c0d0s0 on / type 5.2 (rw, noquota)
   hostname1:/usr/catman on /usr/catman type nfs (ro, soft)
   ```

   (You can also use the `df` command to display the currently mounted file systems.)

6. Test that the manual pages are available by using the `man` command; for example,

```
man fstab
```

If the test succeeded, the text of the `fstab`(4) manual page will print on your screen. If the test remote mount fails, check the default `mount` options (see `mount`(1M)) and Chapter 9, "Troubleshooting." When you have mounted the remote hierarchy successfully, unmount it by typing

```
umount /usr/catman
```

Note: Using the `mount` command from the command line as shown in this section results in a mount that needs to be reentered when the system is rebooted. See "Modifying `/etc/fstab`" for an easier way to remotely mount file systems.

3.6 Modifying `/etc/fstab`

If the `/etc/mount` command is enabled in `/etc/inittab` (see "Enabling the NFS Daemons and `mount` Command"), the `mount` command reads `/etc/fstab` when the system comes up in multi-user mode.

`/etc/fstab` describes the file systems used by the local machine. The file consists of a number of lines such as

```
/dev/dsk/c0d0s0        /            ignore rw        1 1
hostname1:/usr/catman /usr/catman nfs    ro,soft  2 2
```

Fields are separated by blanks or tabs; a number sign (#) as the first non-whitespace character begins a comment.

- For NFS file systems the syntax for the first field is

  hostname`:`*path*

  hostname is always terminated by a colon (`:`), and *path* is a file system mount-point or directory hierarchy below root.

- The second field is the local mount-point. In the example above it must be identical to the remote hierarchy pathname, so that the local `man` command will continue to work. In most cases the

local mount-point can have any pathname you choose. The permissions on that mount-point must be at least as open as the permissions on the remote hierarchy.

- The third field is the type of file system. This may be `4.2`, `5.2`, `nfs`, or `ignore`. If this field is specified as `ignore`, the entire line is ignored. This feature allows you to keep remote mounts that you do not want to mount routinely in `/etc/fstab` when you boot the system.
- The fourth field contains `mount` options. See Table 3-1, `fstab`(4) in *A/UX Programmer's Reference*, and `mount`(1M) in *A/UX System Administrator's Reference* for more information.
- The fifth field is the dump level (used by `dump.bsd`(1M)).
- The last field is the `fsck` pass number. (The value of this field is not used on remote file systems.)

The order of the entries in `/etc/fstab` is important because `fsck`, `mount`, and `umount` process the file sequentially; file systems must appear after file systems in which they are mounted. See `fstab`(4) in *A/UX Programmer's Reference* for more information.

Table 3-1. NFS mount options

NFS mount option	Description
`rw`	Read/write (default). Allow write requests to the remote file system.
`ro`	Read-only. Write-protected hierarchies must be mounted read-only, or errors will occur when access times are updated, even if an explicit write request has not occurred.
`hard`	Retry request until the server responds (default). If a process is accessing remotely mounted data when a server goes down, the process will hang until the server returns, when it will complete as

NFS mount option	Description
	it normally would.
`soft`	Return an error if the server doesn't respond. If a file system is mounted read-only, this option allows the client system to keep running by using only local data when a server system goes down. However, if a process is accessing remotely mounted data when the server goes down, using this option may result in loss of data. It is recommended for use with the `ro` option.
`intr`	Allows most commands to be interrupted when a process is accessing remotely mounted data and a server goes down. This can be used to prevent a process from hanging when the server goes down.
`fg`	If the first mount attempt fails (for example, if the server's mount daemon does not respond), retry the mount attempt in foreground (default). If a client system attempts to mount a remote file system whose server is inaccessible, the client will appear to hang during the mount attempt until the server returns, when the mount attempt will complete as it normally would.
`bg`	If the first mount attempt fails, retry the mount attempt in the background. If the server system is inaccessible, the mount attempt will continue in the background, allowing the client system to be used with local data.
`noauto`	This option is only appropriate when specified in the `/etc/fstab` file. When you specify this option on a line in `/etc/fstab`, that entry is ignored by a `mount -a` command.
`retry=`*n*	Set the number of mount failure retries to *n*

NFS mount option	Description
	(default=1). `mount` attempts each request *n* times before giving up.
`rsize=`*n*	Set the read buffer size to *n* bytes (the default is set by the kernel). The number of bytes in a read request can be set with the `rsize` option.
`wsize=`*n*	Set the write buffer size to *n* bytes (the default is set by the kernel). The number of bytes in a write request can be set with the `wsize` option. The buffer size varies on different types of machines; for example, if the server is a VAX, `wsize=2048` is required.
`timeo=`*n*	Set the NFS timeout to *n* tenths of a second (default=7). Once the file system is mounted, each NFS request made in the kernel waits *n* tenths of a second for a response. If no response arrives, the timeout is multiplied by 2 and the request is retransmitted.
`retrans=`*n*	Set the number of NFS retransmissions to *n* (default=4). When *n* retransmissions have been sent with no reply, a `soft` mounted file system returns an error on the request and a `hard` mounted file system retries the request.
`port=`*n*	Set the server IP port number to *n* (default=`NFS_PORT`). `NFS_PORT` is defined in `<nfs/nfs.h>`. See RFC 960 in "Related RFCs" in Appendix C for more information about IP port numbers.

3.7 Mounting a file system specified in `/etc/fstab`

Once you have modified `/etc/fstab` to include the NFS file systems you want to mount on this machine, those file systems will be mounted whenever you boot the machine. To mount the remote file systems from the command line, enter

```
mount -atv nfs
```

This command mounts the hierarchies specified in `/etc/fstab`. (Note that in "Enabling the NFS Client Daemons," you started a similar command in `/etc/inittab` on the client machine. This automatically mounts all NFS file systems when you boot the client.)

If `mount` is called with only a directory or file system name or file system type, it looks in `/etc/fstab` for entries that match the argument. Thus, if you have the above entry for `/usr/catman` in `/etc/fstab`, you can also enter the command

```
mount /usr/catman
```

If the `mount` program finds no matching entries, it prints an error message; otherwise, it executes the mount indicated in `/etc/fstab`. See `mount`(1M) for more information.

Chapter 4
Adding Yellow Pages Service

Contents

1. Introduction 1

2. Overview of the Yellow Pages 1
   2.1 If you do not use the Yellow Pages 2
   2.2 Masters, slaves, and clients 2
   2.3 Default Yellow Pages maps 3
      2.3.1 Map names and format 4
   2.4 Yellow Pages domains 7
   2.5 Yellow Pages daemons 8

3. Installing the Yellow Pages on the master server 9
   3.1 Setting the domain name 9
   3.2 Creating a global `/etc/passwd` 10
   3.3 Creating a global `/etc/group` 12
   3.4 Checking the default files in `/etc` 12
   3.5 Creating a netgroup file (optional) 12
   3.6 Creating the maps: `ypinit -m` 15
   3.7 Starting the Yellow Pages daemons 16

4. Installing the Yellow Pages on a client system 17
   4.1 Setting the domain name 18
   4.2 Modifying `/etc/passwd` 18
   4.3 Modifying `/etc/group` 20
   4.4 `/etc/hosts` and the other database files 21
   4.5 Starting the Yellow Pages client daemon 22

5. Testing Yellow Pages access 23
   5.1 Summary of Yellow Pages access policies 23

Figures

Figure 4-1. Yellow Pages maps 4
Figure 4-2. Yellow Pages domains 8

Tables

Table 4-1. A/UX short map names 5
Table 4-1. A/UX short map names (cont.) 6
Table 4-2. Map nicknames 7

Chapter 4

Adding Yellow Pages Service

1. Introduction

The A/UX version of the Yellow Pages is based on Release 3.0 of the Sun Microsystems software. This chapter explains how to set up the Yellow Pages service to distribute essential administrative information such as the information typically contained in the `/etc/passwd` file.

"Overview of the Yellow Pages" defines the terms used in this chapter and provides useful information about the implementation of Yellow Pages.

"Installing the Yellow Pages on the Master Server" describes how to make a networked system the Yellow Pages master server. This involves starting daemons, setting a domain name, making global copies of the files used to generate the Yellow Pages databases (maps), and creating the Yellow Pages maps.

"Installing the Yellow Pages on a Client System" describes how to set up a system to query the Yellow Pages instead of local files. This involves starting daemons, setting the same domain name as used on the master server, and modifying the local files.

> *Note:* For example purposes only, this chapter describes setting up a master server and client on a two-system network. In practice, you should also set up at least one backup Yellow Pages server (a "slave server"). See Chapter 5, "Adding Systems to a Network," for information on setting up a slave server.

2. Overview of the Yellow Pages

The Yellow Pages are a collection of programs that generate, maintain, and distribute databases (maps). They are generally used to distribute

administrative information such as password, group, and hosts databases across the network, and they generate these maps by default. See "Default Yellow Pages Maps."

Library routines such as `getpwent`(3C) and `getgrent`(3C) have been rewritten to take advantage of the Yellow Pages. If you import binaries that were created in a non-vnode environment that call such routines, you may have to relink these files for them to work correctly with the Yellow Pages. See Appendix C, "Additional Reading," for a complete description of modified library system calls and subroutines. See also "Modifying `/etc/passwd`" and "Modifying `/etc/group`," for a description of how `getpwent`(3C) and `getgrent`(3C) access the Yellow Pages.

See `ypfiles`(4) for details on the Yellow Pages implementation.

2.1 If you do not use the Yellow Pages

If you choose not to use the Yellow Pages, the procedure for bypassing the software implementation is quite simple: do not enable the daemons or modify the files as described in this chapter.

Remember that if you are using NFS, user IDs must be checked and made unique to each user on all machines on the network, and group IDs must be made consistent across systems. The Yellow Pages's default use is to distribute this information from one source file, facilitating the process of ensuring that they are kept consistent and unique.

2.2 Masters, slaves, and clients

For each Yellow Pages domain, you choose one machine as the Yellow Pages **master server**. On this system, there will be a global copy of the default `/etc` files, and all subsequent modification of these files (for example, to add hosts or users to the network) must occur on the master server only.

> *Note:* If maps are created or changed on slave server machines instead of master server machines, the Yellow Pages update algorithm will be broken.

See "Default Yellow Pages Maps" for a list of the default files and a

description of how the maps are created on the master server.

Because client systems (running the client daemon) will hang if the Yellow Pages maps are not available and a process attempts to retrieve information from the Yellow Pages, you need to maintain redundant Yellow Pages information. To do this, designate other machines as Yellow Pages servers; these are called **slave servers** because they can receive map changes only through the master server. At steady state, it doesn't matter which Yellow Pages server process answers a client request; the answer will be the same all over. This allows multiple servers per network and gives Yellow Pages service a high degree of availability and reliability because if one server becomes unavailable, other servers will take its place. It is very important that the Yellow Pages service be replicated on at least one slave server. Otherwise, if the Yellow Pages server becomes overloaded or goes down, no one on the network will be able to log in to their machines.

The Yellow Pages maps are then distributed to any process that requests them from a Yellow Pages **client** machine, that is, any machine that is running the `ypbind` daemon to access the Yellow Pages information. For example, when a process needs to verify user information (such as ownership of a file), that process issues a remote procedure call (RPC) to the Yellow Pages to obtain the information it requires. The Yellow Pages then retrieves the information from the first available master or slave server. Because the Yellow Pages uses a standard set of access procedures to hide the details of where data is stored, the particular system from which a client process receives information may change at any time.

2.3 Default Yellow Pages maps

The Yellow Pages maps are created automatically by a shell script named `/etc/yp/ypinit` on the master server. After you have set the domain name on the master Yellow Pages server, the `ypinit` command uses it to name a subdirectory of `/etc/yp` that will contain the Yellow Pages maps.

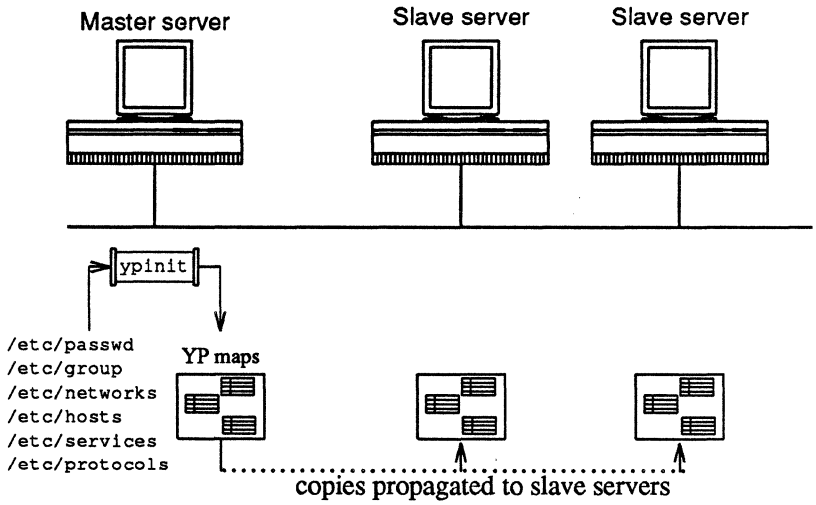
By default, the `/etc/yp/ypinit` script creates maps from `/etc/hosts`, `/etc/passwd`, `/etc/group`, `/etc/networks`, `/etc/services`, `/etc/protocols`, `/etc/ethers`, and `/etc/netgroup`. The `/etc/netgroup`

file can be used to define groups of users that should be permitted to remotely mount certain file systems (see "Creating a Netgroup File (Optional)," later in this chapter).

The `ypinit` script calls a low-level A/UX utility `makedbm`, which converts the information in the ASCII files to `dbm`(3X) database format (a set of keys and associated values). This format is described in `dbm`(4) in *A/UX Programmer's Reference* and "Using `make` on the Default Yellow Pages Maps" in Chapter 7.

Figure 4-1 shows the `ypinit` program generating `dbm`-format maps, which are then propagated to Yellow Pages slave servers.

Figure 4-1. Yellow Pages maps



2.3.1 Map names and format

Sun Microsystems wrote the original implementation of the Yellow Pages for Sun workstations, which are BSD-based and therefore have a 255-character filename limit. A/UX has a 14-character filename limitation. Therefore, this implementation uses "short names" for the map files that reside in the `'domainname'` subdirectory of `/etc/yp` (that is, a subdirectory whose name is the output of the

domainname command). All the long names (and nicknames, where appropriate) work as usual; only the filenames change, and this should not affect the user.

A/UX uses a file named /etc/yp/names.map to establish the correspondence between the full map names and the "short names," as shown in Table 4-1.

Table 4-1. A/UX short map names

Full map name	A/UX short name
group.bynumber	grp.nr
group.byname	grp.nm
group.bygid	grp.g
group.byuid	grp.u
group.byaddr	grp.ad
passwd.bynumber	pwd.nr
passwd.byname	pwd.nm
passwd.bygid	pwd.g
passwd.byuid	pwd.u
passwd.byaddr	pwd.ad
hosts.bynumber	hst.nr
hosts.byname	hst.nm
hosts.bygid	hst.g
hosts.byuid	hst.u
hosts.byaddr	hst.ad
networks.bynumber	ntw.nr
networks.byname	ntw.nm
networks.bygid	ntw.g
networks.byuid	ntw.u
networks.byaddr	ntw.ad
services.bynumber	svc.nr
services.byname	svc.nm
services.bygid	svc.g
services.byuid	svc.u
services.byaddr	svc.ad
protocols.bynumber	ptc.nr
protocols.byname	ptc.nm

Table 4-1. A/UX short map names (cont.)

Full map name	A/UX short name
protocols.bygid	ptc.g
protocols.byuid	ptc.u
protocols.byaddr	ptc.ad
group.time	grp.tm
passwd.time	pwd.tm
hosts.time	hst.tm
netgroup	netg
netgroup.byuser	netg.us
netgroup.byhost	netg.hs
netgroup.time	netg.tm
networks.time	ntw.tm
protocols.time	ptc.tm
services.time	svc.tm
ypmaps	ypmaps
ypservers	ypsrvs
ypdomains	ypdoms
ethers.byaddr	e.ad
ethers.byname	e.nm
rpc.bynumber	rpc.nm
mail.aliases	m.a

The `makedbm` program adds the filename suffixes `.pag` or `.dir` to the A/UX short name for each map name.

For your convenience, nicknames may be used for the longer map names with `ypcat`, `ypmatch`, and `ypwhich`. For example, the command

```
ypcat passwd
```

has exactly the same effect as

```
ypcat passwd.byname
```

and vice versa. The nickname-to-map name correspondence is given by `ypcat -x` and applies to both BSD (long) and A/UX (short)

names (see Table 4-2).

Table 4-2. Map nicknames

Full map name	Map nickname
`passwd.byname`	`passwd`
`group.byname`	`group`
`networks.byaddr`	`networks`
`hosts.byaddr`	`hosts`
`protocols.bynumber`	`protocols`
`services.byname`	`services`

2.4 Yellow Pages domains

When you bring up a system on the network running an NFS kernel for the first time, you are prompted to enter a domain name. When you enter a name in response to the prompt, the name is added as the second field in `/etc/HOSTNAME`. (You can change the system's domain name by editing the second field in `/etc/HOSTNAME` and rebooting.)

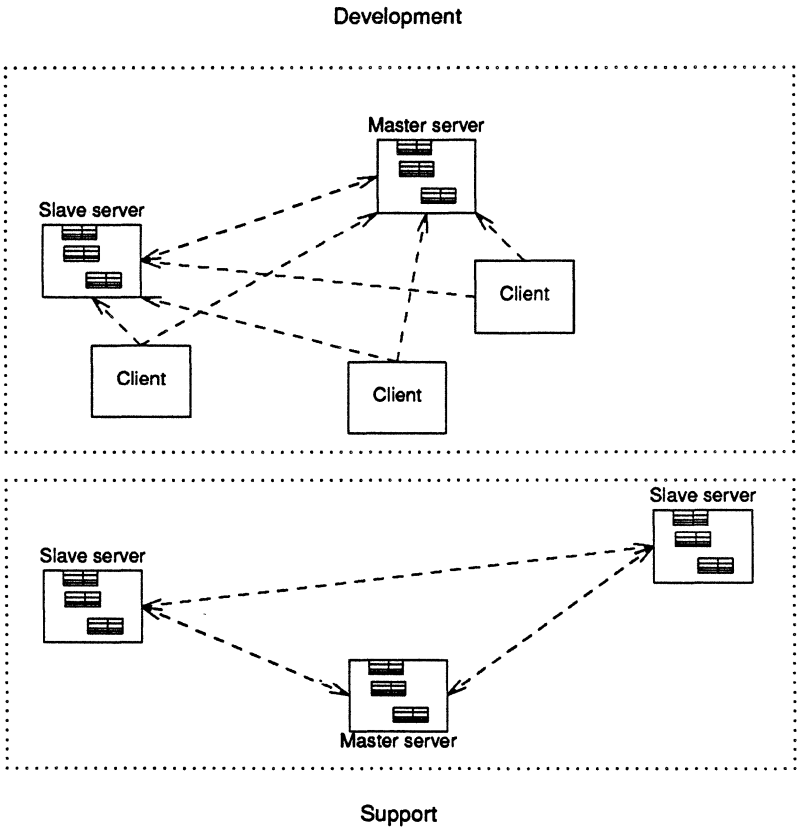
A domain name is required on the master server to create the maps and on the clients to retrieve data from the Yellow Pages. When you are installing the Yellow Pages on the master server, the `ypinit` script uses the domain name as the name of a subdirectory it creates in `/etc/yp`, where the maps are stored. In this sense, a **domain** is a named set of Yellow Pages maps.

Domains are also used to refer collectively to a group of hosts. In this sense, all hosts in a domain use the same domain name and access the same set of Yellow Pages maps. Each domain has at least one master server, and clients from one domain cannot access information from another domain. You can define a new domain for a group of hosts to increase security on those hosts.

Figure 4-2 shows a network with two Yellow Pages domains named "Development" and "Support." The master and slave servers both support copies of the Yellow Pages maps and respond to remote procedure calls (RPCs) to retrieve information from the Yellow Pages. The arrows show RPCs retrieving information. Clients and servers do not "bind" to a particular server, so each RPC request may be

answered by any of the servers. Note that when a server requests information from the Yellow Pages it does not necessarily answer its own request, and that clients from one domain can never retrieve information from servers in another domain.

Figure 4-2. Yellow Pages domains



2.5 Yellow Pages daemons

There are two daemons in the Yellow Pages system:

- The `ypserv` daemon, which serves the Yellow Pages databases to querying processes. This daemon must run on each Yellow Pages server machine.
- The `ypbind` daemon, which issues an RPC call to retrieve information from the Yellow Pages maps. This daemon must run on all machines that use Yellow Pages services, both servers and clients.

3. Installing the Yellow Pages on the master server

It is usually safer to bring up new programs in single-user mode, and this is recommended for bringing up the Yellow Pages. To bring up the master Yellow Pages server, complete the following steps. The subsections that follow explain the steps in more detail.

1. If the domain name set in the second field of `/etc/HOSTNAME` is not correct, edit the file to correct it and reboot to set the correct domain name.
2. Create an `/etc/passwd` file that contains entries for all users on the network.
3. Create an `/etc/group` file that contains entries for all groups on the network.
4. Check that all default files in `/etc` are up-to-date.
5. Run `ypinit -m` to create the Yellow Pages databases.
6. Start the Yellow Pages daemons.
7. Reboot the system.

3.1 Setting the domain name

1. Open the `/etc/HOSTNAME` file by using a text editor. To change the domain name, modify the second field. For example, if the file contains

   ```
   hostname1  apple
   ```

   change `apple` to the domain name you have chosen. This will also be the name of the subdirectory of `/etc/yp` that contains

the Yellow Pages maps.

2. Reboot A/UX. (The system does not read this file until you reboot.)

3.2 Creating a global `/etc/passwd`

When creating a global password file on the Yellow Pages master server, the goal is to make each user ID unique to one user and consistent across the network. This condition is required for remote file access.

Copy the password files from other machines on the network; for example,

```
cd /etc
rcp hostname2:/etc/passwd passwd.2
rcp hostname3:/etc/passwd passwd.3
rcp hostname4:/etc/passwd passwd.4
rcp hostname4:/etc/passwd passwd.5
...
```

After you have copied all the password files, concatenate them together with the local `/etc/passwd`, by using the command

```
cat passwd* > global.passwd
```

Then use a text editor to open `global.passwd`, and edit it as follows:

1. Save the system entries from the local password file; these should be the first 10 or 12 entries. It is important that these remain in the same order and are not modified. You may want to write them to a separate file while you are modifying the rest of this file and read them back in again when you are finished.
2. Delete all other system entries (from password files on other systems).
3. When the file contains only user entries, sort the file (from `vi`, you can use the `:%!sort` command).
4. Eliminate duplicate user entries. When you add Yellow Pages to an existing network, you must be sure that each user has a unique

user name and UID. At this point you may run across instances where the same UID is assigned to different users on different systems or different UIDs are used for the same user on different systems. In this case, make a record of the entries you delete. You will need to give these users unique network-wide user names and UIDs.

5. Check the file and make sure that no users share a UID; change the user names and UIDs, if necessary.
6. When you have modified `passwd.global` to contain only one user entry for each user, read in the file containing the local system entries (see step 1 of this procedure) to the top of the `/etc/passwd` file.
7. The password file must also contain the following entries:

```
daemon:xxxxxxxxxxxx:1:1::/:
nobody:xxxxxxxxxxxx:65534:65534:NFS generic user:/:
/tmp:/bin/noshell
```

(the `nobody` entry is on one line in `/etc/passwd`).

If these lines do not exist, add them below the system entries you just read in. The "daemon" entry allows file-transfer utilities to work and is required for propagating the Yellow Pages database to the slave servers. Note that earlier entries in `/etc/passwd` will mask later ones with the same UID, so make sure that no earlier entry has UID 1. See "Yellow Pages Security Issues" in Chapter 7 for information on the `nobody` entry and on the global password file on the Yellow Pages master server.

When you have completed this process, save the local password file and move the global password into `/etc/passwd`:

```
cp /etc/passwd /etc/passwd.local
cp /etc/passwd.global /etc/passwd
```

If you changed any UIDs, GIDs, or other environment values, write down the machines where you made these changes and then delete the extraneous `passwd.`*n* copies. Be sure to notify users whose UIDs and GIDs you have modified, because these users will not be able to

access their files and directories until you perform the procedure described in the following paragraph.

For example, if you have changed a user's UID on a system, you need to run `chown` on all his files or he will not be able to access them. If the user's previous UID was 400 on this system and you changed it to 125, enter

```
find / -user 400 -exec chown 125 "{}" \;
```

See "Yellow Pages Security Issues" in Chapter 7 if you want to keep a small `/etc/passwd` file on the master server.

3.3 Creating a global `/etc/group`

You should now perform a similar procedure on the `/etc/group` file to obtain an `etc/group` file that is consistent across your network.

3.4 Checking the default files in `/etc`

The rest of the Yellow Pages default files (see "Default Yellow Pages Maps" earlier in this chapter) should already be complete, but check to make sure they are.

3.5 Creating a netgroup file (optional)

Netgroups are network-wide groups of machines and users defined in the `/etc/netgroup` file on the master Yellow Pages server (see `netgroup`(4) for a description of file format and definition of lines and fields). These groups are used for permission checking during remote mount, login, remote login, and remote shell.

The master Yellow Pages server uses `/etc/netgroup` to generate three Yellow Pages maps in the `/etc/yp/`*domain-name* directory: `netg`, `netg.us`, and `netg.hs`. If you do not have an `/etc/netgroup` file, these maps will be zero byte files.

The Yellow Pages map `netg` contains the basic information in `/etc/netgroup`. The two other Yellow Pages maps contain a more specific form of the information to speed the lookup of netgroups given the host or user.

The programs that consult the Yellow Pages netgroup maps are `login`(1) for user classifications if it encounters netgroup names in

/etc/passwd; mountd(1M) for machine classifications if it encounters netgroup names in /etc/exports; rlogin(1) and remsh(1) for both machine and user classifications if they encounter netgroup names in /etc/hosts.equiv or .rhosts.

In this sample /etc/netgroup file, the first field is the netgroup name. The following three fields in the parentheses indicate the host name, user name, and domain name. Any of the three fields can be empty. An empty field signifies a wild card. Thus

```
universal (,,)
```

defines a group to which everyone belongs. Field names that begin with something other than a letter, digit, or underscore (such as "-") work in precisely the opposite fashion. For example, in the following /etc/netgroup file apple is the domain name. Consider the following entries:

```
justmachines   (h7,-,apple)
justpeople     (-,holly,apple)
```

The machine h7 belongs to the group justmachines in the domain apple, but no users belong to it. Similarly, the user holly belongs to the group justpeople in the domain apple, but no machines belong to it.

Here is another sample /etc/netgroup file:

```
#
# Engineering: Everyone has a machine except eric.
# The machine 'h3' is used by all of hardware.
#
engineering hardware software
hardware (h1,alan,apple) (h2,beth,apple) (h3,-,apple)
software (h4,chris,apple) (h5,deborah,apple) (-,eric,apple)
#
# Marketing: Time-sharing on h6
#
marketing   (h6,fran,apple) (h6,greg,apple) (h6,dan,apple)
#
# Others
#
allusers (-,,apple)
allhosts (,-,apple)
```

Based on this sample, the users are classified into groups as follows:

Group	Users
`hardware`	`alan, beth`
`software`	`chris, deborah, eric`
`engineering`	`alan, beth, chris, deborah, eric`
`marketing`	`fran, greg, dan`
`allusers`	(every user in the Yellow Pages map `passwd`)
`allhosts`	(no users)

And here is how the machines are classified:

Group	Hosts
`hardware`	`h1, h2, h3`
`software`	`h4, h5`
`engineering`	`h1, h2, h4, h5, h3`
`marketing`	`h6`
`allusers`	(no hosts)
`allhosts`	(all hosts in the map hosts)

3.6 Creating the maps: `ypinit -m`

To generate the Yellow Pages maps on the master server, enter

```
cd /etc/yp
ypinit -m
```

The `ypinit` program prompts you for information and generates the Yellow Pages maps from `/etc/passwd`, `/etc/group`, `/etc/networks`, `/etc/hosts`, `/etc/services`, `/etc/protocols`, and, if it exists, `/etc/netgroup`.

The `ypinit -m` command displays the following prompt:

```
Do you want this procedure to quit on nonfatal errors?
```

(We recommend that you press `y` for yes.)

The second inquiry made by the `ypinit` program is

```
At this point we have to construct a list
of the hosts that will run YP servers.
"hostname1" is the list of YP server hosts.
Please continue to add the names of the other
hosts, one per line, and when you are finished
with the list, type CTRL-d.
```

If you have this information ready, enter the host names one to a line, and when you are finished press CONTROL-d. These host names will be included in the `ypsrvs` map. If you need to add or delete slave servers later, reconstruct the `ypsrvs` map as described in "Adding a Yellow Pages Slave Server" in Chapter 5. If you do not yet know which machines will run as Yellow Pages slave servers, just press

CONTROL-d.

The program then lists the server machines again and asks if the list is correct. If it is, press `y` to begin the actual generation of maps (which could take several minutes).

3.7 Starting the Yellow Pages daemons

On the master server, start the Yellow Pages daemons `ypserv`, `ypbind`, and `yppasswd`. These daemons are invoked from `/etc/inittab`. To invoke the `ypserv`, `ypbind`, and `yppasswd` daemons:

1. Edit `/etc/inittab` to change the status fields (shown in bold) of the following lines from

   ```
   nfs1:2:off:/etc/ypserv
   nfs2:2:off:/etc/ypbind
   ```

   to

   ```
   nfs1:2:wait:/etc/ypserv
   nfs2:2:wait:/etc/ypbind
   ```

2. The `yppasswdd` daemon is invoked on the master Yellow Pages server only. This daemon automatically updates the master's password database when users change their passwords with the `yppasswd` command. (See "Redefining the `passwd` Command" in Chapter 7 for ways to make this change transparent to the user.)

   To invoke the `yppasswdd` daemon, add the following line to `/etc/inittab`:

   ```
   nfs9:2:wait:/usr/etc/rpc.yppasswdd /etc/passwd -m passwd
   ```

   This causes a `make` (the `-m` flag) and a `yppush` to occur for every password change on the network. This may cause too much performance degradation on a large network. To avoid this, you can modify this line to read

   ```
   nfs9:2:once:/usr/etc/rpc.yppasswdd /etc/passwd
   ```

   This will cause `/etc/passwd` on the master server to be updated whenever your system enters the multi-user state, but

will not modify and propagate the Yellow Pages map. Therefore, you need to create a `crontab` entry to perform periodic updates automatically. Rather than having a separate `crontab` entry for each Yellow Pages map, you can group commands to update several maps in a shell script. Examples are in the following files in `/etc/yp`:

```
ypxfr_1d
ypxfr_2d
ypxfr_1h
```

(That is, mnemonically, "yp transfer once per day," "yp transfer twice per day," and "yp transfer once per hour.")

The disadvantage of this approach is that changes made to the master server are not immediately seen throughout the net. That is, database maps may not always be current on the slave servers, and thus users may access incorrect information.

See "Yellow Pages Security Issues" in Chapter 7 for changes you can make to the above `yppasswdd` invocations to keep a restricted-access `passwd` file on the master server.

If you modified `/etc/inittab` in single-user mode, enter multi-user mode now; `init` will read your changes when your system enters multi-user mode. Otherwise, reboot your system to start the daemons.

With the database created and the daemons running, the machine is now the Yellow Pages master server. You can use the `ps` command now to verify that the Yellow Pages daemons, `ypserv`, `ypbind`, and `rpc.yppasswdd`, are running. See Chapter 7 for more information about managing the Yellow Pages on the master server.

4. Installing the Yellow Pages on a client system

This chapter assumes that the Yellow Pages client is named `hostname2`. To make `hostname2` a client system, bring the system up in single-user mode and complete the steps explained in this section. The subsections that follow explain the steps in more detail.

1. If the domain name set in the second field of `/etc/HOSTNAME` is not correct, edit the file to correct it and reboot to set the correct domain name. *The domain name on a client system must be the same as the domain name on the server.*
2. Modify the client's local `/etc/passwd` and `/etc/group` files.
3. Start the Yellow Pages daemons.
4. Reboot the system.

4.1 Setting the domain name

1. Open the `/etc/HOSTNAME` file by using a text editor. To change the domain name, modify the second field; for example, if the file contains

   ```
   hostname2  apple
   ```

   change `apple` to the domain name you have chosen. *This must be the same domain name used on the master server.* This will also be the name of the subdirectory of `/etc/yp` that contains the Yellow Pages maps.
2. Reboot A/UX. (The system does not read this file until you reboot.)

4.2 Modifying `/etc/passwd`

When a user program calls the `getpwent` (get password entry) library routine, it reads the local `/etc/passwd` file just as it always has, but it interprets + entries in the password file to mean "interpolate entries from the Yellow Pages database." If the system is a Yellow Pages client (running `ypbind`), a remote procedure call retrieves the entry from the Yellow Pages server. (For example, if you wrote a simple program using `getpwent` to print out all entries in your password file, it would print out a virtual password file that would include all local entries and all entries interpolated from the Yellow Pages database.)

The `/etc/passwd` file is processed sequentially. When a user entry is found, processing stops, so earlier entries mask later ones, and local entries mask Yellow Pages entries. A sample Yellow Pages client's

/etc/passwd file looks like this:

```
root:wAm0Y4lEnf6:0:1:God:/:/bin/csh
nobody:*:65534:65534::/:
daemon:*:1:1::/:
operator:VyZr6V9:333:20:sys op:/usr2/operator:/bin/csh
+joe:EBd4YJUeS45DA:0:0:Joe Smith:/users/joe:/bin/ksh
+::0:0:::
```

These entries include the following:

`root`
: So that `root` can log in and so that all `root` passwords are not identical on all NFS systems.

`nobody`
: To prevent superuser access to remotely mounted files. See "Yellow Pages Security Issues" in Chapter 7 for more information.

`daemon`
: To allow file-transfer utilities to work.

`operator`
: To allow a dump operator log in.

primary users
: The primary users of the machine should be included to save the overhead of Yellow Pages access when these users are logging in and to allow these users to log in even if the Yellow Pages are unavailable. Note, however, that their login sessions will still be interrupted if the Yellow Pages are unavailable and the local `/etc/group` file accesses the Yellow Pages. See "Modifying `/etc/group`."

`+`
: Lines beginning with a plus sign call the Yellow Pages. In the sample file above, the last line,

  ```
  +::0:0:::
  ```

  tells the library routines to use the Yellow Pages rather than give up the search.

Note: Do not put such a line in the master server's `/etc/passwd` file, because it allows unrestricted access to the entire domain.

There are four styles of + entries:

+
Used alone, + means to insert the entire contents of the Yellow Pages password file at that point.

`+::0:0:::`
Means to consult the Yellow Pages for any other entries.

+name
Means to insert the entry (if any) for *name* from the Yellow Pages at that point.

If a "+ *name*" entry has non-null fields; for example,

```
+joe:EBd4YJUeS45DA:0:0:Joe Smith:/users/joe:/bin/ksh
```

where the fields are

name:*password*:*UID*:*GID*:*comment*:*directory*:*shell*

this tells the library routines to use the Yellow Pages, but to allow non-null password, comment, directory, or shell fields to override what is contained in that field of the Yellow Pages entry. However, the user UID and GID fields in such an entry are automatically taken from the Yellow Pages.

+@*netgroup*
Means to insert the entries for all members of *netgroup* at that point.

4.3 Modifying `/etc/group`

When a user program calls the `getgrent` (get group entry) library routine, it reads the local `/etc/group` file just as it always has, but it interprets + entries in the password file to mean "interpolate entries

from the Yellow Pages database." If the system is a Yellow Pages client (running the `ypbind` daemon), an RPC to the server retrieves the interpolated entry.

However, unlike the `passwd` file, processing of `/etc/group` does not stop when a match is found; the entire file is searched for all possible groups to which a user might belong. If you include a + entry in `/etc/group`, it searches the entire Yellow Pages group map for every `getgrent` call. Thus, if you use the Yellow Pages for group verification, you can abbreviate the local `/etc/group` file to a single line:

```
+:
```

This escape sequence forces all translation of group names and group IDs to be made via the Yellow Pages service.

> *Note:* When you use this escape sequence, the group name and group ID verification process searches the entire Yellow Pages group database to find the correct group and check whether a user belongs to that group. If all Yellow Pages servers on the network are down, the local system will be unavailable to all users (even primary users).
>
> If you have the adequate redundancy that using a number of slave servers provides, this should not be a problem. An alternative solution is not to use Yellow Pages service at all, by keeping a local `/etc/group` file with no + entries. In this case, you should still make sure the group IDs do not conflict with group IDs on the rest of the network, in case you decide later to use the Yellow Pages.

4.4 `/etc/hosts` and the other database files

The `/etc/hosts` file does not require any modification on the Yellow Pages client systems. When a user program calls the `gethostbyname` or `gethostbyaddr` library routines, if the system is a Yellow Pages client (running `ypbind`) these routines may go to the Yellow Pages before consulting the local `/etc/hosts` file.

If you modify the local `/etc/hosts`, remember that it must contain entries for the local host and the local loopback name. These are accessed at boot time, before the Yellow Pages are available.

For example, the minimal local `/etc/hosts` file for `hostname2` would be

```
127.0.0.1   loop lo loo localhost  #loopback
128.8.1.2   hostname2          #local hostname and address
```

The remaining database files (`/etc/networks`, `/etc/protocols`, `/etc/services`, and `/etc/netgroup` if it exists) are treated identically to `/etc/hosts`; if the Yellow Pages are running, the local files are not consulted. These files should be left as they are in case the Yellow Pages become unavailable for any reason.

4.5 Starting the Yellow Pages client daemon

The `ypbind` daemon is invoked from `/etc/inittab`. To invoke this daemon:

1. Edit `/etc/inittab` to change the status fields (shown in bold) of the following line from

   ```
   nfs2:2:off:/etc/ypbind
   ```

   to

   ```
   nfs2:2:wait:/etc/ypbind
   ```

2. If you are in single-user mode (and have not previously entered multi-user mode), enter

   ```
   init 2
   ```

   Otherwise, reboot A/UX.

Note: Make sure there is a Yellow Pages server before putting the client machine in multi-user mode. Otherwise, the machine is likely to hang if no Yellow Pages server is available while `ypbind` is running. Use the `ypwhich` command, explained in the next section, to test for this condition.

5. Testing Yellow Pages access

To see whether the client system recognizes the Yellow Pages server, from the client machine enter

```
ypwhich
```

The `ypwhich` command returns the host name of the server machine that is currently being used to access the Yellow Pages. The response in the case of the two-system net in this chapter should be

```
hostname1
```

The command

```
ypcat passwd
```

prints the values in the Yellow Pages `passwd` map, which is normally served off the remote host.

To test that all is working as it should be, perform the ultimate test:

1. Log in to the master Yellow Pages server machine, and create a normal user entry in `/etc/passwd`. Be sure to create a password for this user entry, or else the entry won't be served by the Yellow Pages.
2. Propagate this entry to all slave server machines with the commands

   ```
   /etc/yp/yppush passwd.byname
   /etc/yp/yppush passwd.byuid
   ```

3. Log in as this user on any machine running Yellow Pages service within the master server's domain. (For a proper test, this should be any machine but the master server.) If the login is successful, the Yellow Pages are up and running.

5.1 Summary of Yellow Pages access policies

`/etc/passwd`

The local file is always consulted. If there are + or - entries, the Yellow Pages password map is also consulted; otherwise, the Yellow Pages are unused. Local entries mask

analogous Yellow Pages entries.

`/etc/group`
The local file is always consulted. If there are + or - entries, the Yellow Pages group map is also consulted; otherwise, the Yellow Pages are unread.

`/etc/hosts`
The local file is consulted when booting. After that, the Yellow Pages are used before the local file is checked.

`/etc/networks`
The local file is never consulted. The Yellow Pages are used instead.

`/etc/services`
The local file is never consulted. The Yellow Pages are used instead.

`/etc/protocols`
The local file is never consulted. The Yellow Pages are used instead.

`/etc/netgroup`
The local file is never consulted. The Yellow Pages are used instead.

`/etc/ethers`
The local file is never consulted. The Yellow Pages are used instead.

`/etc/hosts.equiv`
The local file is always consulted. If it contains + or - entries whose arguments are netgroups, the Yellow Pages netgroup map is consulted; otherwise the Yellow Pages are unused. See "Yellow Pages Security Issues" in Chapter 7.

`$HOME/.rhosts`
The local file is always consulted. If it contains + or - entries whose arguments are netgroups, the Yellow Pages netgroup map is consulted; otherwise the Yellow Pages are unused. See "Yellow Pages Security Issues" in Chapter 7.

Chapter 5
Adding Systems to a Network

Contents

1. Introduction 1

2. Adding an A/UX system to the network 1

3. Adding NFS systems without the Yellow Pages 5
   - 3.1 Adding NFS servers 5
   - 3.2 Adding an NFS client 6

4. Adding systems by using the Yellow Pages 8
   - 4.1 Adding a Yellow Pages client system 8
   - 4.2 Adding a Yellow Pages slave server 10

Chapter 5

Adding Systems to a Network

1. Introduction

This chapter describes how to add systems to a network. If the network is not using the Yellow Pages, you need to manually copy files across the network to inform systems about each other. If the network is using the Yellow Pages, many of the changes can be made on the master server only.

"Adding an A/UX System to the Network" describes how to bring a system up on a network that is not using the Yellow Pages.

"Adding NFS Systems without the Yellow Pages" describes how to make NFS systems functional after adding them to the network. The procedure is the same as described in Chapter 3; it is simply summarized here.

"Adding Systems by Using the Yellow Pages" describes how to use the Yellow Pages to add hosts to the network. If the network is using the Yellow Pages, you will save a considerable amount of time by making changes to the master server `/etc/hosts` database and using the Yellow Pages to inform the network about the new host. This section also describes how to bring up a Yellow Pages slave server and client.

2. Adding an A/UX system to the network

> *Note:* When adding a system to an existing network, you should make all the required software changes before enabling the daemons at reboot. It is recommended that you make the changes in single-user mode to ensure that the new system is not up on the net before other systems know about it.

If you are adding a system to a network that is not using the Yellow

Pages, you should choose a system that is already up on the network as a host machine for network files such as `/etc/hosts`, `/etc/hosts.equiv` (if desired), and `/etc/networks`. This section assumes that the host system is named `hostname1` and the new system is named `hostname3`.

1. Log in to `hostname1` (a system already on the network), and modify its `/etc/hosts` (and `/etc/hosts.equiv` if you wish) to include an entry for the new system `hostname3`.

2. Go back to `hostname3` and bring it up in single-user mode. Edit `/etc/inittab` to change the status field (shown in bold) on the following lines from

   ```
   nfs0:2:off:/etc/portmap
   net4:2:off:/usr/etc/in.routed
   net5:2:off:/usr/etc/in.rwhod
   net9:2:off:/etc/inetd
   ```

   to

   ```
   nfs0:2:wait:/etc/portmap
   net4:2:wait:/usr/etc/in.routed
   net5:2:once:/usr/etc/in.rwhod
   net9:2:respawn:/etc/inetd
   ```

   and save the new version of `/etc/inittab`. Also check `/etc/servers` as described in "Modifying `/etc/inittab`" in Chapter 2.

3. On `hostname3`, make sure `root` can execute the installation scripts by setting the permissions:

   ```
   chmod 755 /etc/newunix
   chmod 755 /etc/install.d/*
   ```

   Then enter either

   ```
   /etc/newunix bnet
   ```

   or, if this machine will also run NFS,

   ```
   /etc/newunix nfs
   ```

4. Reboot and enter `launch newunix` from the stand-alone shell (or run the `autoconfig` command to build a new kernel and reboot), and answer the prompts that appear on your screen during reboot as described in Chapter 2, "Establishing a Two-System Network."

5. When you are back in single-user mode on `hostname3`, use the `ftp` utility to copy `/etc/hosts` from `hostname1` to `/etc/hosts` on `hostname3`. (The `ftp` utility is used instead of `rcp` because, on a properly configured network, `rcp` does not allow `root` to make file transfers across the network.)

   Enter the command

   ```
   ftp 128.8.1.1
   ```

   (where `128.8.1.1` is the Internet address of the host system, `hostname1`). This will print something like the following:

   ```
   Connected to hostname1.
   220 hostname1 FTP server ready.
   Name (hostname1:root): ordinary-user
   ```

   After the `Name` prompt, type the user name you would normally use on `hostname1` when you are not logged in as `root` and press RETURN.

   Note: The `ftp` utility reads the file `/etc/ftpusers` to determine which users to exclude from the system. It is normal practice to ship systems with a `root` entry in this file, even though `ftp` will always ask for a password if one is present in `/etc/passwd`.

   After the `Password` prompt,

   ```
   Password (hostname1:ordinary-user):
   ```

   type the password and press RETURN (the password does not echo on the screen). The `ftp` utility then prints something like the following:

```
331 Password required for ordinary-user.
230 ordinary-user logged in.
ftp>
```

6. At the `ftp>` prompt, enter the command

   ```
   get /etc/hosts
   ```

   Note: Because you are logged in as `root` on `hostname3` and because `/etc/hosts` always has read permission for everyone, `ftp` will successfully make the file transfer to `/etc/hosts` on `hostname3`.

   The `get` command will print something like the following:

   ```
   200 PORT command successful.
   150 Opening data connection for /etc/hosts
       (128.008.001.001) (2147 bytes).
   226 Transfer complete
   local:/etc/hosts remote:/etc/hosts
   2247 bytes received in 0.04 second
   ftp>
   ```

7. When the file transfer is complete and you see the `ftp>` prompt again, enter

   ```
   quit
   ```

8. Log in to your other network systems as `root` and in like manner use `rcp` or `ftp` to copy the updated `/etc/hosts` file to those systems that need to communicate with the new system.

If you want to allow users on the new system access to remote systems without supplying a password, you may also modify the remote systems' `/etc/hosts.equiv` files. This can also be accomplished by individual users creating a `.rhosts` file in their login directories on the remote system.

3. Adding NFS systems without the Yellow Pages

This section describes how to add NFS servers and NFS clients that do not use Yellow Pages to your network. The procedure is the same as described in Chapter 3; the information is simply summarized here for your convenience.

3.1 Adding NFS servers

A large disk or multiple disks on a system are probably the most important consideration in choosing a system to be an NFS file server. Another prime consideration should be the system's reliability. Most remotely mounted read-write hierarchies are mounted with the default `hard` option. The `hard` option attempts to insure data integrity by hanging the process on the client machine when the remote server crashes. When the remote server returns to service, the client process resumes where it left off. Unfortunately, a crashed server machine and hung processes on a client machine result in effective down time for users on both machines.

After you have brought a new system up in the network by following the procedure described in "Adding an A/UX System to the Network," you can export its file systems by following these steps:

1. Bring the system up in single-user mode, edit `/etc/inittab` to change the status field (shown in bold) in the following line from

   ```
   nfs3:2:off:/etc/nfsd 4
   ```

   to

   ```
   nfs3:2:wait:/etc/nfsd 4
   ```

   and save the new version of `/etc/inittab`.

2. Edit `/etc/exports` to include a line exporting the root file system (if the system supports a single disk) and any specific hierarchies (if desired) followed by the appropriate list of hosts. For example,

   ```
   /  # export to everyone
   ```

3. Bring the system up to multi-user mode by typing

```
init 2
```

4. Check that the NFS server daemons are running by typing

```
ps -ef | grep nfsd
```

The response should be

```
root  81  1  0 01:27:17 ?    0:02 /etc/nfsd 4
root  82  1  0 01:27:17 ?    0:02 /etc/nfsd 4
root  83  1  0 01:27:17 ?    0:02 /etc/nfsd 4
root  84  1  0 01:27:17 ?    0:02 /etc/nfsd 4
```

(The numbers in each column may be different on your system).

3.2 Adding an NFS client

After you have brought a new system up on the network by following the procedure described in "Adding an A/UX System to the Network," (and an NFS server has an updated copy of `/etc/hosts` so it can communicate with the new system), first check that the system can communicate with a file server:

```
ping hostname1
```

(where `hostname1` is a server). If the response is similar to

```
64 bytes from 128.8.1.1: icmp_seq=0. time=16. ms
64 bytes from 128.8.1.1: icmp_seq=1. time=16. ms
64 bytes from 128.8.1.1: icmp_seq=2. time=16. ms

(interrupt)

----hostname1 PING Statistics----
3 packets transmitted, 3 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 16/16/16
```

then check the export permissions on `hostname1`:

```
showmount -e hostname1
```

If `hostname1` exports to everyone, you can follow the steps in this section to mount a hierarchy on the new system. Otherwise, log in as `root` on `hostname1` and add the new system's host name to the export list in `/etc/exports`; that is, add the new system's host

name, separated by white space from the other host names, to the desired lines in this file. For example, to export to the new system `hostname3`, `/etc/exports` on `hostname1` might look like the following:

```
/           hostname2 hostname3
/usr/catman hostname2 hostname3
```

Follow these steps to add your new system to the network as an NFS client.

1. Bring the new system up in single-user mode and enter

   ```
   grep nobody /etc/passwd
   ```

   The response should be

   ```
   nobody:xxxxxxxxxxxxx:65534:65534:NFS generic user:
   /tmp:/bin/noshell
   ```

   (This should appear on one line on the screen.) If this is not the response, modify `/etc/passwd` to include this line.

2. Edit `/etc/inittab` to change the status field (shown in bold) in the following lines from

   ```
   nfs4:2:off:/etc/biod 4
   nfs8:2:off:/etc/mount -at nfs >/dev/syscon 2>&1
   ```

   to

   ```
   nfs4:2:wait:/etc/biod 4
   nfs8:2:once:/etc/mount -at nfs >/dev/syscon 2>&1
   ```

   and save the new version of `/etc/inittab`.

3. Edit `/etc/fstab` to add an entry for mounting the remote hierarchy as described in Chapter 3, "Initializing NFS."

4. Check that the local mount-point you specified in `/etc/fstab` exists, is empty, and has permissions at least as open as the remote hierarchy. If the local mount-point does not exist, create it with

```
mkdir /dir-name
```

and check the permissions as described in Chapter 3.

5. Bring the system up to multi-user mode by typing

```
init 2
```

6. Check that the appropriate daemons are running by typing

```
ps -ef | grep biod
```

The response should be

```
root  81  1  0 01:27:17 ?    0:02 /etc/biod 4
root  82  1  0 01:27:17 ?    0:02 /etc/biod 4
root  83  1  0 01:27:17 ?    0:02 /etc/biod 4
root  84  1  0 01:27:17 ?    0:02 /etc/biod 4
```

(The numbers in each column may be different on your system).

4. Adding systems by using the Yellow Pages

This section describes how to add a Yellow Pages client to your network. After adding a Yellow Pages client, follow the directions in "Adding a Yellow Pages Slave Server" if you want to make this system a slave server.

4.1 Adding a Yellow Pages client system

When you are adding a Yellow Pages client system, you can make most of the system file changes on the master Yellow Pages server alone.

1. Log in as `root` on the master Yellow Pages server.
2. Edit `/etc/hosts` on the master Yellow Pages server, and add an Internet address and host name (and if you wish, a nickname for the system). For example, if the new system's name is `hostname5`, use

```
128.8.1.5  hostname5 h5
```

3. Update the Yellow Pages maps by typing

```
cd /etc/yp
make
```

4. Log in as `root` on the new system. Unless you have already entered the correct domain name in response to the `domainname` prompt when rebooting with the NFS kernel, use a text editor to enter the domain name of the master server in the second field of `/etc/HOSTNAME`. (The field delimiters are blanks or tabs.)

   ```
   hostname5      apple
   ```

   For example, change `apple` to the same domain name used by the master Yellow Pages server. (If you do not know the domain name, enter

   ```
   domainname
   ```

   on the master server.)

5. Modify the client's `/etc/passwd`, `/etc/group`, and `/etc/hosts` files as described in Chapter 4, "Adding Yellow Pages Service."

6. Edit `/etc/inittab` to change the status fields (shown in bold) of the following line from

   ```
   nfs2:2:off:/etc/ypbind
   ```

   to

   ```
   nfs2:2:wait:/etc/ypbind
   ```

7. Reboot the machine.

   Note: There must be at least one `ypserv` process running on the network before you reboot the client system or the machine will hang during the boot process.

8. Create home directories for users who should have access to the new system.

To see whether the client system recognizes the Yellow Pages server, enter

```
ypwhich
```

The `ypwhich` command returns the host name of the server machine that is currently being used to access the Yellow Pages.

4.2 Adding a Yellow Pages slave server

After you have brought a system up as a Yellow Pages client on the network, perform the following steps to make it a slave server for the Yellow Pages:

1. On the master Yellow Pages server, log in as `root` and make sure the network is working by trying a `ping` or `telnet` command.
2. Generate a new set of Yellow Pages maps by typing

   ```
   cd /etc/yp
   ypinit -m
   ```

   `ypinit -m` prints the following prompt:

   ```
   Do you want this procedure to quit on nonfatal errors?
   ```

   (We recommend that you press `y` for yes.)

   The second inquiry made by the `ypinit` program is

   ```
   At this point we have to construct a list
   of the hosts that will run YP servers.
   "hostname1" is in the list of YP server hosts.
   Please continue to add the names of the other
   hosts, one per line, and when you are finished
   with the list, type CTRL-d.
   ```

   Type the name of the new system that will be a Yellow Pages slave server; for example,

   ```
   hostname3
   ```

   followed by CONTROL-d.

This procedure modifies the `ypsvrs` database on the master server. After you have thus informed the master server that there is a new

slave server,

1. Log in as `root` on the new slave server system (`hostname3`), and make sure the network is working by trying a `ping` or `telnet` command.
2. Check the second field of `/etc/HOSTNAME` to make sure the system is in the same domain as the master server. If these fields are not identical on both systems, use a text editor to enter the master's domain name in the second field of the slave's `/etc/HOSTNAME`. (The field delimiters are blanks or tabs.)
3. Check that `/etc/passwd` contains the following entry for `daemon`:

   ```
   daemon:xxxxxxxxxxxx:1:1::/:
   ```

   If this line is not in the slave system's `/etc/passwd`, edit the password file to include this line.
4. Type the commands

   ```
   cd /etc/yp
   ypinit -s hostname1
   ```

   where `hostname1` is the master server (or another Yellow Pages slave server). The `ypinit -s` command sets up the Yellow Pages database on the slave Yellow Pages server machine.
5. Edit `/etc/inittab` to change the status fields (shown in bold) of the following lines from

   ```
   nfs1:2:off:/etc/ypserv
   nfs2:2:off:/etc/ypbind
   ```

   to

   ```
   nfs1:2:wait:/etc/ypserv
   nfs2:2:wait:/etc/ypbind
   ```
6. Reboot A/UX and come up to multi-user mode.

The machine is now a Yellow Pages slave server.

Chapter 6
Network Design Issues

Contents

1. Introduction 1

2. Network design considerations 1
   2.1 Routing and forwarding 2
   2.2 Subnets 8
      2.2.1 Subnets and broadcasting 11
   2.3 Internet domains 12
      2.3.1 Enabling the resolver software 13

Figures

Figure 6-1. Two separate networks 3

Figure 6-2. An Internet forwarder system 4

Figure 6-3. Subnets 9

Tables

Table 6-1. `/etc/hosts` on example networked systems 7

Table 6-2. `/etc/NETADDRS` on example networked systems 8

Table 6-3. `/etc/hosts` on example subnets 10

Table 6-4. `/etc/NETADDRS` on example subnets 11

Table 6-5. Subnets and broadcast addresses 12

Chapter 6

Network Design Issues

1. Introduction

This chapter describes various design decisions in setting up a larger network and tells how to use the software supported by A/UX to set up an Internet forwarder machine, configure subnets (supporting only Macintosh II computers), and add an A/UX system to a larger network that is running the Internet domain software.

2. Network design considerations

If you are configuring a large network, you need to consider the physical limitations of the network based on electrical parameters, the number of hosts connected, the total length of the cable, and the physical location of hosts (that is, whether the geographical distribution allows direct Ethernet connections between all hosts on the network). In a large organization, such as a university or a company with more than one building, you need a way of communicating between multiple Ethernet cables. To do this, you can

- Obtain a distinct Internet network number for each cable and use Internet forwarders to route between networks. (Note that if these networks connect to the outside world, the internal routing details are propagated to other systems that have no use for this information, resulting in unnecessarily large Internet routing tables.) This is described under "Routing and Forwarding."
- Use a single network number and partition the host address space by assigning subnet numbers to the local area networks. In this case, the internal division is transparent to the outside world. See "Subnets."

In a large network you may also want to use the BIND (Berkeley Internet Name Domain) host name and Internet address server, which has been modified slightly in A/UX to accommodate the Yellow Pages.

The domain software is part of the 4.3BSD BIND distribution. See Appendix B, "Name Server Operations Guide for BIND," and "Related RFCs" in Appendix C for documentation about Internet domains.

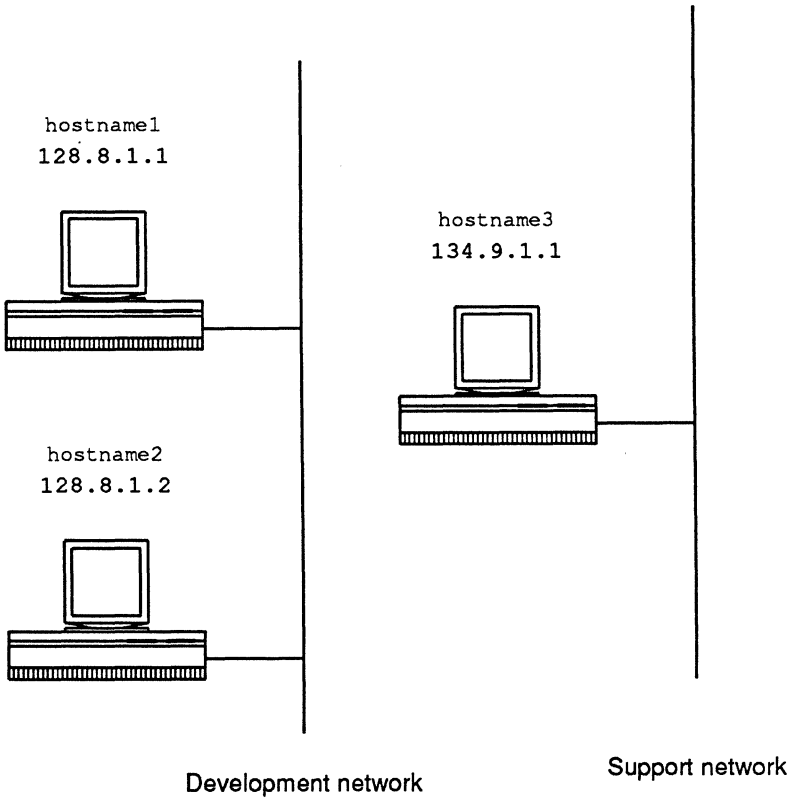
2.1 Routing and forwarding

Some organizations have more than one Ethernet and obtain a separate network number on each cable. In this situation, you can configure the separate networks to talk to each other by setting up Internet forwarder systems. An Internet forwarder is connected to two separate networks and routes packets between them.

> *Note:* If you are setting up multiple Ethernets, all systems should run the `routed` daemon.

This section describes setting up a Macintosh II as an Internet forwarder between two networks. For the purpose of illustration, assume the conditions shown in Figure 6-1. The two networks are called the Development and Support networks and use the network numbers `128.8` and `134.9`, respectively. The Development net currently supports two Macintosh II computers named `hostname1` and `hostname2`. The forwarder system will be `hostname2`. The Support net supports one Macintosh II named `hostname3`. The Internet addresses for each machine are as shown.

Figure 6-1. Two separate networks

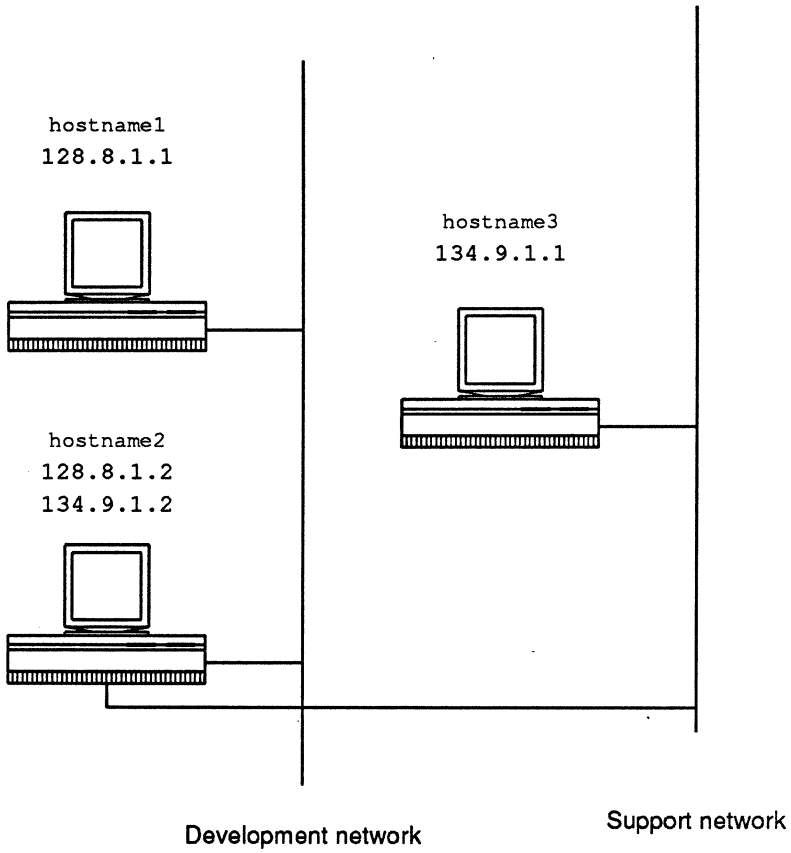


To establish a forwarder between the two networks, you need to choose one system on one network (Development network in Figure 6-1) to forward traffic to the other network (Support network in Figure 6-1), and make the other network able to communicate with the forwarder system.

The forwarder system (`hostname2`) needs two Ethernet cards (one card connected to each network) and two Internet addresses. Systems on the Development network communicate with `hostname2` by using an address that begins with the network number `128.8`, and

systems on the Support network communicate with `hostname2` by using an address that begins with the network number `134.9`. This is shown in Figure 6-2.

Figure 6-2. An Internet forwarder system



To set up `hostname2` as an Internet forwarder system, first install a second Ethernet card and cable that connects the new card to the second network. Then do the following:

1. Log in as `root` on `hostname2`.
2. Enter

   ```
   sync; sync; reboot
   ```

   When the automatic reboot screen comes up and starts counting down, use the mouse to click Cancel or press the COMMAND key and a period (`.`) to cancel the reboot. This puts you in the Stand-Alone SHell (`sash`).
3. Enter

   ```
   launch newunix
   ```

   When the system reboots, be sure to enter `y` to answer the question "Check the root file system?" (If A/UX reports that it has fixed the root file system and then it reboots, cancel the reboot from `sash` and then reenter `launch newunix`).

   As the system is booting, it notices the new Ethernet card and prompts you for the Internet address, broadcast address, and netmask associated with the new card. For example

   ```
   2 Ethernet card(s) installed
   ae1:  Please enter an Internet address:
   ae1:  Please enter an Internet Broadcast address:
   ae1:  Please enter a netmask [none]:
   ```

   Note: You may use the DELETE key to edit typing mistakes *before* pressing RETURN.
4. In response to these prompts, enter the system's address on the Support network. For example,

   ```
   134.9.1.2
   ```

   Next, enter the broadcast address used on the Support network. For example,

   ```
   134.9.255.255
   ```

   Note that the broadcast address on the second Ethernet interface

on `hostname2` (the forwarder system) must match `hostname3`'s broadcast address. Before entering the broadcast address for the new Ethernet card, check the broadcast address on `hostname3` by doing one of the following:

- ☐ View `hostname3:/etc/NETADDRS` (the broadcast address and netmask are in the third and fourth fields in this file),
- ☐ or enter

  ```
  ifconfig ae0
  ```

  on `hostname3`. This will tell you the broadcast address and netmask.

5. Finally, supply the netmask used on the Support network; for example,

   ```
   0xffff0000
   ```

All systems on both networks must be running the routing daemon `/etc/in.routed`, which should be set to `wait` in `/etc/inittab`, or static routes must be installed manually by using `route`(1M). However, only `hostname2` will broadcast routing updates. The forwarder system broadcasts to the separate networks to let them know about the presence of the other network and routes packets between the two networks; if `hostname1` wants to contact `hostname3`, it communicates through `hostname2`.

Note: For each of the Ethernet interfaces on the forwarder system, the broadcast address must agree with the one used by the other system(s) on that network.

Now you need to make `hostname3` able to talk to `hostname2` on the Support network. To do this:

1. Log in as `root` on `hostname3`, and open `/etc/hosts` with a text editor.
2. Comment out or delete the current entry for `hostname2`, and add an entry for `hostname2`'s second Internet address. For

example,

```
134.9.1.2  hostname2 h2          # macII
```

Table 6-1 shows the `/etc/hosts` files on the three systems.

Table 6-1. `/etc/hosts` on example networked systems

hostname1

```
#Development network
128.8.1.1 hostname1 # Tue Oct 15 01:45:51 PDT 1987
128.8.1.2 hostname2             # macII

#Support network
134.9.1.1 hostname3             # macII
127.0.0.1  loop lo localhost    # local loopback
```

hostname2

```
#Development network
128.8.1.1  hostname1            # macII
128.8.1.2  hostname2 # Tue Oct 15 01:45:51 PDT 1987

#Support network
134.9.1.1  hostname3
127.0.0.1  loop lo localhost    # local loopback
134.9.1.2  hostname2 # Tue Oct 15 02:15:06 PDT 1987
```

hostname3

```
#Development network
128.8.1.1 hostname1             # macII

#Support network
134.9.1.2 hostname2             # macII
127.0.0.1  loop lo localhost    # local loopback
134.9.1.1  hostname3  # Tue Oct 15 01:45:51 PDT 1987
```

Note: If you forced the host name and domain name prompt to appear by removing the `/etc/NETADDRS` file, you may now have duplicate entries for this host *at the same Internet address* in `/etc/hosts`. In most cases, these duplicate entries cause no harm. However, if this host is a Yellow Pages master server, be sure to remove these duplicate entries from `/etc/hosts`. Otherwise, inconsistent Yellow Pages behavior may result.

Table 6-2 shows the `/etc/NETADDRS` files on the three systems.

Table 6-2. `/etc/NETADDRS` on example networked systems

hostname1	0	128.8.1.1	128.8.255.255	0xffff0000
hostname2	0	128.8.1.2	128.8.255.255	0xffff0000
	1	134.9.1.2	134.9.255.255	0xffff0000
hostname3	0	134.9.1.1	134.9.255.255	0xffff0000

After you have set up the second network, you should modify `/etc/networks` on all systems (or modify it on the master server and use the Yellow Pages) to contain an entry for the new network. Modifying `/etc/networks` is described in "Other Required System Files" in Chapter 2.

2.2 Subnets

For administrative or technical reasons, you may choose to divide the network into several subnets rather than obtaining several unique network numbers. If you use subnets, the internal routing details are not visible to the outside world.

The subnet code is part of the IP module. It allows the standard "host" field on an Internet address to be split into two parts: a "subnet part" and a "host part", using a 32-bit netmask. The netmask contains binary 1's and 0's. The 1's in the netmask define the "network part" of the Internet address, and the 0's in the mask define the "host part" of the Internet address.

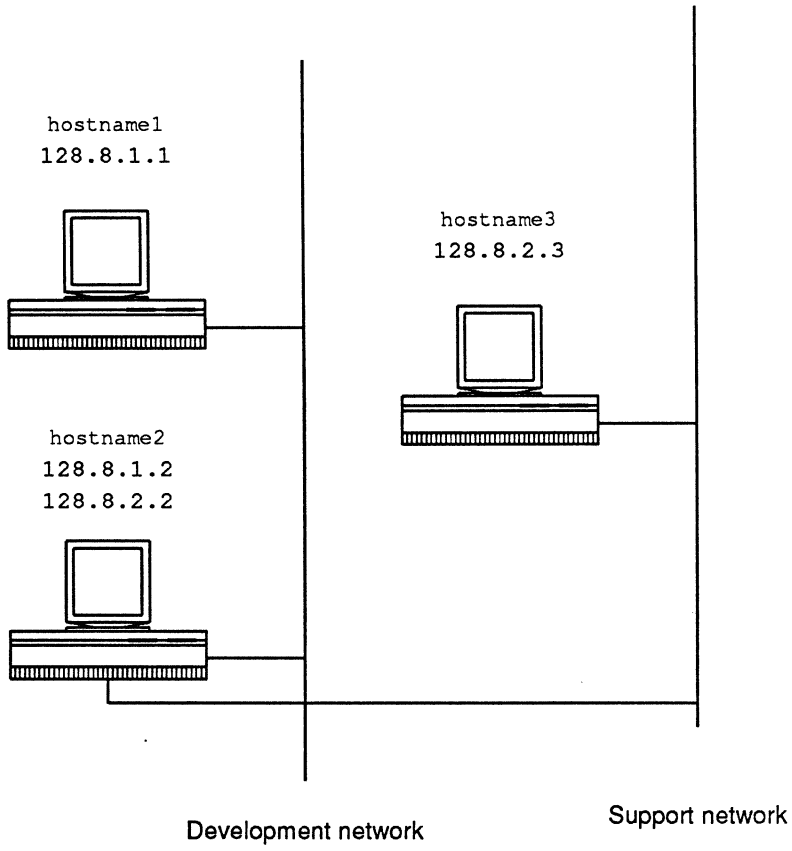
The actual Internet address for a system does not change (and is therefore consistent for the outside world), but the interpretation of the address by the local network is changed.

Because subnets are not visible to the outside world, the netmask cannot reduce the number of bytes in the network number. For example, on a class B network number, a netmask of `0xffff0000` specifies no subnets, because it uses the class B standard of 2 bytes of network number and 2 bytes of host number. To support subnets, you can only *add* significant bits to this "default" network mask. For example, the netmask

```
0xffffff00
```

allows the third byte (the first byte of the host field of the Internet address) to be used as the subnet number because it adds a byte to the network part of the Internet address.

Figure 6-3. Subnets



The procedure for setting up subnets is the same as described under "Routing and Forwarding" except for the addressing scheme. Figure 6-3 shows three machines on two networks at a site that uses the class

B network number

```
128.8
```

All three systems use Internet addresses beginning with `128.8` and a netmask that defines the third byte as part of the network number:

```
0xffffff00
```

In Figure 6-3, `hostname2` is set up as a forwarder system with two Ethernet cards and a connection to both networks.

Table 6-3 shows the `/etc/hosts` files on the three subnetted systems.

Table 6-3. `/etc/hosts` on example subnets

hostname1

```
#Development network
128.8.1.1 hostname1 # Tue Oct 15 01:45:51 PDT 1987
128.8.1.2 hostname2             # macII

#Support network
128.8.2.3 hostname3             # macII
127.0.0.1  loop lo localhost    # local loopback
```

hostname2

```
#Development network
128.8.1.1  hostname1            # macII
128.8.1.2  hostname2 # Tue Oct 15 01:45:51 PDT 1987

#Support network
128.8.2.3  hostname3
128.8.2.2  hostname2            # macII
127.0.0.1  loop lo localhost    # local loopback
128.8.2.2  hostname2 # Tue Oct 15 02:15:06 PDT 1987
```

hostname3

```
#Development network
128.8.1.1 hostname1             # macII

#Support network
128.8.2.2 hostname2             # macII
127.0.0.1  loop lo localhost    # local loopback
128.8.2.3  hostname3  # Tue Oct 15 01:45:51 PDT 1987
```

Table 6-4 shows the `/etc/NETADDRS` files on the three subnetted systems.

Table 6-4. `/etc/NETADDRS` on example subnets

hostname1	0	128.8.1.1	128.8.1.255	0xffffff00
hostname2	0	128.8.1.2	128.8.1.255	0xffffff00
	1	128.8.2.2	128.8.2.255	0xffffff00
hostname3	0	128.8.2.3	128.8.2.255	0xffffff00

2.2.1 Subnets and broadcasting

Some daemons broadcast frequently (for example, the `rwho` daemon). For this reason, if many systems receive broadcast packets with the wrong broadcast address, Ethernet storms with high collision and network traffic rates can result, generally decreasing the efficiency of the network.

In the examples in the preceding section, if both subnets support only Macintosh II computers, you could leave the broadcast address as

```
128.8.255.255
```

on both subnets. Then any broadcast would reach every system on either subnet. See "Related RFCs" in Appendix C for more information.

If the network supports different types of systems, however, you should limit their broadcasts to their own subnet. This is the method employed in the examples above. Table 6-5 shows how different netmasks may be used with broadcast addresses.

Table 6-5. Subnets and broadcast addresses

Network class	Netmask	Broadcast address
A	`0xffff0000`	*n1*`.`*n2*`.255.255`
B	`0xffffff00`	*n1*`.`*n2*`.`*n3*`.255`
C	`0xfffffff0`	*n1*`.`*n2*`.`*n3*`.15`

Note that, in the example, for a class C network number the netmask uses 4 bits for the subnet number and the same broadcast address. If you use a subnet mask of `0xfffffff0` with a class C network, you may have total of 14 subnets with 14 hosts on each (`0` and all `1`'s have special meaning, and so they're avoided as network, subnet, and host numbers). So, for a class C network number of

```
192.11.1
```

and a netmask of

```
0xfffffff0
```

you would have Internet addresses as follows:

```
192.11.1.0x11          (host 1 on subnet 1)
192.11.1.0x12          (host 2 on subnet 1)
192.11.1.0x21          (host 1 on subnet 2)
...
```

Note: A/UX software allows the `0x` (hexadecimal) notation. Software and hardware from other manufacturers may require the decimal equivalent of the hexadecimal number.

2.3 Internet domains

The A/UX software supports Internet domains. For detailed information about Internet domain name servers, see `named`(7) in *A/UX System Administrator's Reference*, `resolver`(4) in *A/UX Programmer's Reference*, Appendix B, "Name Server Operations Guide for BIND", and the documentation related to Internet domains in "Related RFCs" in Appendix C.

2.3.1 Enabling the resolver software

If you are adding an A/UX system to an existing 4.3BSD network running the BIND distribution, the libraries that query the domain name server are already in place. See `resolver`(3) for more information.

For the system to query a specific Internet name server instead of broadcasting to any local name server, you must create a configuration file named `/etc/resolve.conf` that specifies the desired name server. The required configuration is described in `resolver`(4) in *A/UX Programmer's Reference*.

On the Macintosh II, host name or address queries from a user process are sent first to the name server. If the information is not found, the query is sent to the Yellow Pages. If the information is not found there, the query is sent to the local `/etc/hosts` file. If all three sources do not include the requested information, the query fails.

Chapter 7
Network Management

Contents

1. Network security 1
   1.1 B-NET security issues 1
      1.1.1 A friendly network 1
         1.1.1.1 Remote login as `root` 2
      1.1.2 A more secure network 2
      1.1.3 Increasing network security 3
   1.2 NFS security issues 3
      1.2.1 No superuser access over the network 3
         1.2.1.1 Changing the mode 4
         1.2.1.2 Changing ownership 5
         1.2.1.3 Allowing root access 6
   1.3 Yellow Pages security issues 8
      1.3.1 `/etc/passwd` 8
      1.3.2 The Yellow Pages and `/etc/hosts.equiv` 8

2. Network user administration 9
   2.1 Networked systems without the Yellow Pages . . . 10
   2.2 Using the Yellow Pages 10
      2.2.1 Adding an entry to the master server's password file 10
      2.2.2 Adding an entry to the master server's group file 11
      2.2.3 Updating the Yellow Pages maps 11
      2.2.4 Removing a user from the network 11
      2.2.5 Redefining the `passwd` command 12

3. B-NET administration 13
   3.1 Backing up networked systems 13
      3.1.1 Backing up to a remote backup device 14

3.1.2 Restoring files from a remote backup device 15

4. NFS administration 17
4.1 System performance 17
4.1.1 Memory 17
4.1.2 Improving NFS throughput 17
4.1.3 Warning: Do not remotely mount important executable files 18
4.2 Using links to optimize disk space 18
4.2.1 Hard links and symbolic links 18
4.2.2 Creating and removing links 19
4.2.3 Using NFS and symbolically linked directories 20

5. Yellow Pages administration 23
5.1 Server system performance 23
5.2 Updating the Yellow Pages maps 23
5.2.1 Using **make** on the default Yellow Pages maps 23
5.3 Propagation of a Yellow Pages map 24
5.3.1 Periodically by **cron** 24
5.3.2 By **ypserv** 24
5.3.3 Interactively by a user 25
5.4 Adding a slave server 25
5.4.1 Modifying the master server's databases 25
5.4.2 Initializing databases on the slave server 27
5.5 Using a new master server 28
5.5.1 Old master out of service 28
5.5.2 Old master in service 30

6. A network mail system 31
6.1 Starting the **sendmail** daemon 32
6.2 **sendmail** implementation notes 34

Chapter 7

Network Management

This chapter discusses various topics related to managing your network. These topics include how to increase the security of your network, how to set up a `sendmail` file on your network, and how to back up files across the network. Administration management for a B-NET network, NFS network, and NFS network with the Yellow Pages is also discussed.

1. Network security

As a network administrator, you determine the level of security on your network. This section describes how to make your system a friendly network, how to make your system more secure, and how to address specific security issues associated with B-NET, NFS, and NFS with Yellow Pages networks.

1.1 B-NET security issues

Several degrees of network security are possible with the networking software.

1.1.1 A friendly network

If your network is "friendly" and you can trust all the users, you can allow relatively open access to networked machines by specifying all the hosts on the network in the `/etc/hosts` and `/etc/hosts.equiv` files of each machine.

The `/etc/hosts.equiv` file establishes a host "equivalence": Users with entries in the `/etc/passwd` files for two machines can use the `rlogin` command to log in remotely from one system to another without using a password. For example, the `/etc/hosts.equiv` file on `hostname1` might include

```
hostname2
hostname3
hostname4
```

In this case, all non-root users who have a legitimate login account on `hostname1` and on the specified machines can log in remotely to `hostname1` from those machines without supplying a password. (See also "The Yellow Pages and `/etc/hosts.equiv`," later in this chapter.)

If user names differ across machines, users may still use `rlogin` by using the `-l` option (see `rlogin`(1N) in *A/UX Command Reference* for more information). Note, however, that `user1` on `hostname2` will be able to log in to `user1`'s account on `hostname1`, even if he or she does not own that account!

Note: Although the `/etc/hosts.equiv` file accepts a user name as the second field on a line, this is not recommended. This would allow the user superuser access to the local system.

1.1.1.1 Remote login as `root`

Unless you have a very open network environment, you should not allow the superuser on one machine to automatically use `rlogin` as superuser on another machine. If you decide to allow this, you can create a `/.rhosts` file on each machine that specifically names host(s) and `root`; for example,

```
hostname1 root
```

Note: Because this further reduces network security, it is not recommended.

1.1.2 A more secure network

To eliminate the open access at the system level, you can remove the system's `/etc/hosts.equiv` file. You can then allow specific users access to their own accounts by installing `$HOME/.rhosts` files in their home directories. The format of this file is identical to that of `/etc/hosts.equiv` but allows the specified users from the

specified hosts to log in to that user's account.

1.1.3 Increasing network security

If you need a secure network system, you may wish to disable the `rlogind`, `remshd`, and `tftpd` daemons, which do not check passwords to authorize users. To do this, edit `/etc/servers` and delete the lines shown in bold:

```
ftp      tcp    /usr/etc/in.ftpd
telnet   tcp    /usr/etc/in.telnetd
shell    tcp    /etc/in.remshd
login    tcp    /etc/in.rlogind
exec     tcp    /usr/etc/in.rexecd
tftp     udp    /usr/etc/in.tftpd
talk     udp    /usr/etc/in.talkd
finger   tcp    /usr/etc/in.fingerd
rpc      udp    /usr/etc/rpc.rstatd   100001   1-2
rpc      udp    /usr/etc/rpc.rwalld   100008   1
rpc      udp    /usr/etc/rpc.mountd   100005   1
```

1.2 NFS security issues

The NFS software currently assumes that you have a "friendly" net; that is, you can trust all of the users attached to your net. This section describes security-related issues.

1.2.1 No superuser access over the network

> *Note:* The subsection "Allowing Root Access" describes how to undo the UID 0/UID –2 mapping to allow root superuser access permissions over the network, but *this is not recommended.* (In `/etc/passwd` the `nobody` entry uses a UID of `65534`—the unsigned representation of –2 in 2's complement notation.)

The following example shows how `root` is treated as another user on remote files:

1. Log in as a user on the local system and use the command

```
mount
```

to see which directories are remotely mounted.

2. Check the permissions on the remotely mounted directory; for example, if `h1` is a remotely mounted hierarchy,

```
$ cd /h1/users/joe
$ ls -ld .
```

This will print something like

```
drwxr-x---  16 joe   prog   512 Mar 13 14:10 ./
```

Note that these permissions do not allow access to "other."

3. Use `su` to switch to `root`:

```
$ su
Password:
```

and repeat the `ls` command:

```
# ls .
#
```

Because UID 0 (`root`) has been mapped to UID –2 (`nobody`), the directory will appear empty. Don't panic; the file system still exists. You are simply denied access to it as `root`.

There are several ways around the problem of no root access on remotely mounted file systems, depending on the security requirements of your network environment. These are described below.

1.2.1.1 Changing the mode

Note: A program that is setuid `root` will not be able to access remote files or directories unless permissions include "other."

You cannot change the mode of files remotely mounted as `root`. To get around this problem, you can use `su` as the owner of the files, or log in to the server machine and change the mode of files or directories on the machine where they reside.

For example, a user `joe` can use `touch` and `chmod` on his own remote mounted files:

```
$ touch /h1/users/joe/test1 /h1/users/joe/test2
$ chmod 777 /h1/users/joe/test1
$ chmod 700 /h1/users/joe/test2
$ ls -l /h1/users/joe/test*

-rwxrwxrwx  1 joe  0 Mar 24 16:12 /h1/users/joe/test1
-rwx------  1 joe  0 Mar 24 16:12 /h1/users/joe/test2
```

However, if you are the superuser, the `test2` 700 mode file will not allow any operations, but the `test1` 777 mode will, and so the date and time should change.

```
$ su
Password:

# touch /h1/users/joe/test1
# touch /h1/users/joe/test2
touch: /h1/users/joe/test2: Permission denied

# ls -l /h1/users/joe/test*

-rwxrwxrwx  1 joe  0 Mar 24 16:14 /h1/users/joe/test1
-rwx------  1 joe  0 Mar 24 16:12 /h1/users/joe/test2
```

1.2.1.2 Changing ownership

As the superuser, you cannot change ownership of remotely mounted files. For example, if you try to use `chown` on a program named `a.out` (which is setuid root) that is located on a remotely mounted file system, you will see something like this:

```
$ chmod 4755 /h1/users/joe/a.out
$ su
Password:
# chown root /h1/users/joe/a.out
a.out: Not owner
```

Because users cannot execute a `chown` command on a file that is setuid root and because `root` is not treated as the superuser on remote access, there are only two ways to change ownership of remote files. You can log in (with `login`, `rlogin`, or `telnet`) to the server machine as `root` and execute a `chown` command there, or you can

use `rcp` to copy the remote file to a file system owned by your machine (for example, `/usr/tmp` should always be on the local machine) and make the change in the copy.

1.2.1.3 Allowing root access

In a very friendly network environment, you may choose to allow root access over the network.

> *Note:* This will compromise system security measures for your network and is not recommended if your network requires secure systems.

The following procedure allows you to use `adb` on the A/UX kernel on a server machine. Note that this works only on a server machine, not on a client.

> *Note:* Be extremely careful when modifying a kernel location. When in doubt, exit with CONTROL-d and do not use the `/W` or `?W` command to write out the changes.

1. On the NFS server, change the value of the kernel variable `nobody`. Login in as root on a running system, and enter

   ```
   adb -k -w /unix /dev/kmem
   ```

   `adb` should respond

   ```
   a.out file = /unix     (COFF format)
   core file  = /dev/kmem
   ready
   ```

   This is normal.

2. Type

   ```
   nobody /D
   ```

   The "`/`" tells `adb` to get (or write) the value of `nobody` from (or to) kernel memory; the `D` says to print the value in long decimal format. The response should be

```
nobody:          -2
```

If `adb` does not respond with these numbers, stop.

- ☐ Press CONTROL-d to exit `adb`.
- ☐ Verify that you have invoked `adb` with the proper flag options and that you have already run `/etc/newunix` and `/etc/autoconfig` (or `launch newunix`) to incorporate the B-NET or NFS modules.
- ☐ Try recreating your kernel with `/etc/newunix` and `/etc/autoconfig` if the problem persists.

3. If it does respond as above, enter

```
nobody/W 0
```

This command changes the value of `nobody` in the running memory image of the kernel. You should see the response

```
nobody:  0xFFFFFFFE  =  0x0
```

When you have completed these steps, the currently running kernel will allow root access to NFS clients.

4. If you want to modify the value of `nobody` on disk image of the kernel (`/unix`) so that it will be in effect each time the kernel is booted, type

```
nobody?W 0
```

instead of

```
nobody/W 0
```

The `?` tells `adb` to get (or write) the value of `nobody` from (or to) the disk image of the kernel (`/unix`).

To check if the change was written out correctly, type

```
nobody ?D
```

`adb` should respond

```
nobody:  0
```

Press CONTROL-d to exit `adb`.

1.3 Yellow Pages security issues

This section discusses specific steps you can take to enhance the security of your network.

1.3.1 `/etc/passwd`

The master server's `/etc/passwd` should contain entries for users who have accounts on all the Yellow Pages client machines. This provides access for a user to all machines that access the Yellow Pages. If you prefer to use a small password file on the Yellow Pages master server to keep access to it restricted, you can install the global password file in a directory other than `/etc` (for example, in `/etc/yp`) and change the line in `/etc/inittab` that invokes the `yppasswdd` daemon on the Yellow Pages master server.

Instead of using the line

```
nfs9:2:once:/usr/etc/rpc.yppasswdd /etc/passwd
-m passwd
```

(on one line in `/etc/inittab`), you can use the following line (assuming you installed the global password file as `/etc/yp/passwd`):

```
nfs9:2:once:/usr/etc/rpc.yppasswdd /etc/yp/passwd
-m passwd DIR=/etc/yp
```

(This should appear on one line in `/etc/inittab`.) This passes a new value for the `DIR` variable to `/etc/yp/Makefile`.

1.3.2 The Yellow Pages and `/etc/hosts.equiv`

The `/etc/hosts.equiv` and `$HOME/.rhosts` files are generally used to allow users access to a machine without providing their passwords. However, you may wish to explicitly *deny* certain groups of users access to a particular Yellow Pages client.

Using the Yellow Pages, you can modify these files to deny access to all users on particular hosts (defined in the Yellow Pages `hosts` map) or all hosts and certain users (defined in the Yellow Pages `netgroups` map).

If a user is entered in both `/etc/passwd` and `$HOME/.rhosts` files, B-NET allows that user to use `rlogin`, `rcp`, or `remsh` without providing a password.

Because the Yellow Pages `passwd` map is global and allows access to most users on the network, you may want to modify `/etc/hosts.equiv` on certain machines to restrict access by groups of users or all users on certain hosts. You may also want to inform users on these restricted machines that they may modify their `$HOME/.rhosts` file to allow access.

To modify these files to use the Yellow Pages, use a plus sign (+) or minus sign (–) and an at sign (@) followed by the host name or netgroup name. The + allows access and the - denies access.

+@*trusted-netgroup*
+@*trusted-host*
–@*distrusted-netgroup*
–@*distrusted-host*

For example, if you don't want the administrative department to be able to log in on `hostname2`, the `/etc/hosts.equiv` file on `hostname2` should contain the line

```
–@admin
```

and the `netgroup` Yellow Pages database should contain an entry for the `admin` netgroup followed by the login names of people in that department. See "Creating a Netgroup File (Optional)" in Chapter 4.

Note: The root `/.rhosts` file controls remote root access to the local machine and should be restricted unless you have a very friendly network environment. It is possible to use the same conventions as `/etc/hosts.equiv` in each user's `.rhosts` file.

2. Network user administration

This section describes the procedure for allowing a new user access to (1) a network that does not use the Yellow Pages and (2) a network that

does use the Yellow Pages service.

2.1 Networked systems without the Yellow Pages

If you have decided not to use the Yellow Pages, the procedure for adding users to a machine on the network is essentially the same as adding a user to a single machine. This procedure (described in *A/UX Local System Administration*) should be done on each machine to which the user needs access. You need to check manually to see that the user's UID and GID are consistent on each machine.

2.2 Using the Yellow Pages

This section describes how to

- Add an entry to the master server's `/etc/passwd` file.
- Add an entry to the master server's `/etc/group` file.
- Update the Yellow Pages maps.
- Remove a user from the network.
- Redefine the `passwd` command.

2.2.1 Adding an entry to the master server's password file

To add an entry to the master server's password file,

1. Log in as `root` on the master server.
2. Use `vipw` to edit the global `/etc/passwd` file. This prevents any user or program from accessing `/etc/passwd` while you are editing it. See `passwd`(4) or "User Administration" in *A/UX Local System Administration* for information about the password file format.
3. Add an entry for the new user. For example, if the user's name is Mr. Garcia and his login name will be `jerry`, add a line such as

   ```
   jerry::714:100:Mr. Garcia:/users/jerry:/bin/sh
   ```

4. Write and exit this file, then assign a password to the user with the command

   ```
   passwd jerry
   ```

Note: You pose a security risk for every machine on the network by not assigning the new account a password before including it in the Yellow Pages maps. Inform the user of the new password, and instruct him or her to change it during the first log in.

The command to change a password in the Yellow Pages is

```
yppasswd
```

See "Redefining the `passwd` Command" for various ways of making the Yellow Pages more transparent to the user.

2.2.2 Adding an entry to the master server's group file

To add an entry to the global group file, add the new user's login name to all appropriate groups in the master server's `/etc/group` file.

If the user is to be a member of groups particular to a specific client, you should also add the user's name to these groups in the client's local `/etc/group` file.

2.2.3 Updating the Yellow Pages maps

To update all the maps that need to be updated:

1. Change directory to `/etc/yp` on the master server.
2. Enter the command

```
make
```

If you have Yellow Pages slave servers on your network, the updated maps will automatically be progagated to those servers listed in the `ypsrvs` map. See "Yellow Pages Administration" for more information on the `ypsrvs` map and adding slave servers.

2.2.4 Removing a user from the network

A/UX creates files and directories by using the user's UID and GID numbers in the master server's global password file and client group files. If you were to simply delete a user from the global password file (and client group files), the user's files would still keep the same UID and GID numbers. If you then assigned these numbers to a new user, this user would inherit ownership of the old user's files.

There are two ways to deal with this problem. Either

- Remove all file and directories belonging to the old user (backing them up if necessary), and then delete the user entry from the global password file and from the client group files.

or

- Deny the old user access to the network by changing the user password field in the global password file to a single asterisk (`*`).

We recommend the second method because you don't have to remove files and you will always know who created those files. (We also recommend that for safety's sake you use `vipw` whenever you edit a password file.)

Whichever method you choose, be sure to remake the Yellow Pages maps:

```
cd /etc/yp
make
```

2.2.5 Redefining the `passwd` command

When a user changes his or her password with the `passwd` command, only the entry in the local client's `/etc/passwd` file is changed. However, if you are using the Yellow Pages to handle user accounts, most user entries are not in the local `/etc/passwd` files but are pulled in from the Yellow Pages with a `+` entry. In this case, the `passwd` command prints the error message

```
Changing password for user.
Permission denied.
```

Thus, the `passwd` command is inappropriate, and the user should use the `yppasswd` command instead.

You may wish to make the `passwd` command inoperable. Here are some ways to do this.

Note: `/bin/passwd` must be owned by `root` and must be mode 4755. Thus, you must be `root` when you copy it to save the permissions.

- Save the `/bin/passwd` command under another name, and symbolically link `/bin/passwd` to `/usr/bin/yppasswd` with the `ln -s` command.
- Save the `/bin/passwd` command under another name (using the `mv` command), and replace it with a shell script that prints

  ```
  Please use yppasswd to change your password.
  ```

  You will need the `/bin/passwd` program to change any local passwords. Therefore, you should rename it to something familiar like `/bin/localpasswd`. Furthermore, be sure to tell all users who have local passwords of this command name to change.
- It is possible to alias `passwd` to `yppasswd` in the `/etc/cshrc` and `/etc/profile` files. For `/etc/cshrc`, use the command

  ```
  alias passwd yppasswd
  ```

  For `/etc/profile`, use

  ```
  passwd(){
       yppasswd
  }
  ```

3. B-NET administration

This section describes how to implement a backup strategy for a networked system. For information on administrating a network mail system, see "A Network Mail System".

3.1 Backing up networked systems

This section describes how to use the utilities `rdump`(1M) and `rrestore`(1M) to back up and restore files resident on a local machine, by using a backup device (tape or disk) attached to a remote machine.

The `rdump` utility backs up a file system by directly accessing the raw file system device (for example, `/dev/rdsk/c0d0s0`). This means that `rdump` will not back up remotely mounted file system hierarchies. These hierarchies will be backed up when `rdump` (or

dump.bsd) is run on the machine where the hierarchy actually resides.

Note: Always back up server machines regularly. Should a Yellow Pages master server machine fail, you will need to create a new master server from a recent backup of Yellow Pages maps and files.

3.1.1 Backing up to a remote backup device

The rdump command allows you to use the network to make routine backups of a local machine by using a remote machine's backup device.

The rdump facility works just like dump.bsd; that is, it copies to magnetic media all files changed after a certain date in the file system. You must explicitly tell rdump to access the remote machine and tape drive with the syntax

/etc/rdump *n*f *host-name*:*device*

where *n* is the dump level and *host-name*:*device* is the device-special file corresponding to the backup device on the remote machine. (The f key is always required to explicitly specify the device-special file on the remote machine.)

For example,

```
rdump 0uf hostname1:/dev/rdsk/c0d0s0
```

The 0 flag option causes the entire file system to be dumped. The u flag option writes the date of the beginning of the dump on the /etc/dumpdates file if the dump completes successfully.

The rdump command creates a server (/etc/rmt) on the remote machine to control the backup device.

Note: If your site uses a custom backup program employing the the stat() system call to traverse file system trees, your program will not handle symbolic links correctly. Changing all stat() calls to lstat() calls and recompiling should solve the problem.

See `rdump`(1M) and `dump.bsd`(1M) and "Backups" in *A/UX Local System Administration* for more information.

3.1.2 Restoring files from a remote backup device

The `rrestore` command is an incremental file system restore program similar to the standard `restore` command. Like `rdump`, `rrestore` creates a server (`/etc/rmt`) on the remote machine to access the remote backup device and restore files locally.

The `rrestore` process works just like standard `restore`, except that you must explicitly tell the program on which machine the remote backup device resides. The syntax is

```
/etc/rrestore -f host-name:device
```

For example,

```
/etc/rrestore -fi hostname1:/dev/rdsk/c0d0s0
```

The `f` flag option tells `rrestore` to use the next argument as the name of the archive. The `i` flag option allows interactive restoration of files from a dump tape.

After reading in the directory information from the tape, `rrestore` provides a shell-like interface that allows the user to easily move around the directory tree and selectively extract files. The commands for this interface are explained below.

The default for commands accepting an optional argument is the current directory.

`ls` [*arg*]
: List the current or specified directory. Entries that are directories are appended with a `/`. Entries that have been marked for extraction are prefixed with an `*`. If the verbose key is set, the inode number of each entry is also listed.

`cd` *arg*
: Change the current working directory to the specified argument.

`pwd` Print the full pathname of the current working directory.

`add` [*arg*]
: Add the current directory or specified argument to the list of files

to be extracted. If a directory is specified, it and all its descendents are added to the extraction list (unless the `h` key is specified on the command line). Files that are on the extraction list are prefixed with an `*` when they are listed by `ls`.

`delete` [*arg*]
Delete the current directory or specified argument from the list of files to be extracted. If a directory is specified, it and all its descendents are deleted from the extraction list (unless the `h` key is specified on the command line). The most expedient way to extract most of the files from a directory is to add the directory to the extraction list and then delete those files that are not needed.

`extract`
Extract from the dump tape all the files that are on the extraction list. `rrestore` will ask which volume the user wishes to mount. The fastest way to extract a few files is to start with the last volume and work toward the first volume.

`setmodes`
Set the owner, modes, and times for all the directories that have been added to the extraction list. Nothing is extracted from the tape. This is useful for cleaning up after a restore has been prematurely aborted.

`verbose`
Set or disable verbose by pressing `v`, which acts as a toggle. When set, the `v` key causes the `ls` command to list the inode numbers of all entries and the `rrestore` command to print information about each file as it is extracted.

`help`
List a summary of the available commands.

`quit`
Exit `rrestore` immediately, even if the extraction list is not empty.

See `rrestore`(1M) and `restore`(1M) and "Backups" in *A/UX Local System Administration* for more information.

4. NFS administration

This section discusses how to administer your NFS system to achieve greater performance, to increase throughput, and to optimize disk space.

4.1 System performance

The first priority of an NFS server is to service I/O requests from client machines. If a file system is exported to several machines, each with multiple users accessing that data, the server machine slows down in the attempt to handle the remote procedure calls. This noticeably affects users on the server machine. Performance on the client will also be affected, but it will not be as noticeable because the client caches data in its own buffers.

4.1.1 Memory

In theory, NFS requires only a small amount of memory greater than the kernel size (approximately 1 MB) to run. In practice, however, system performance degrades if processes have to be paged out more often because of the insufficient memory. CPU-intensive processes will noticeably slow system response time.

> *Note:* For optimal performance, a minimum of 4 MB of memory is recommended for NFS servers.

4.1.2 Improving NFS throughput

The NFS daemon `nfsd` runs on an NFS server to handle client file system requests. The NFS daemon `biod` runs on an NFS client to buffer asynchonous block I/O between the client and server. (It is possible for a machine to be an NFS client and not run `biod`, but this will likely make the client very inefficient.)

If many remote requests are coming in to a server machine, you can run multiple `nfsd` daemons to handle the request bottleneck and thereby increase throughput. Likewise, you can increase the number of `biod` daemons running on a client system to improve its throughput.

Of course, every running process requires some system overhead, and NFS daemons are no exception to this rule. Thus, running too many NFS daemons could conceivably create a bottleneck.

Note: Practical experience has shown that optimal performance is usually obtained by running four `nfsd` daemons on the server and four `biod` daemons on the client. (A server machine may also be an NFS client.) See `nfsd`(1M) and `biod`(1M) in *A/UX System Administrator's Reference*.

4.1.3 Warning: Do not remotely mount important executable files

NFS clients should not remotely mount executable files required to boot, talk to the network, or perform indispensable tasks. These files include `/unix`, `newunix`, `/nextunix` and the `/bin` and `/etc` hierarchies.

4.2 Using links to optimize disk space

Because client machines may have a small amount of disk space, you may want to organize its contents using links. A **link** is a pointer to a file that allows the file to appear in two or more places within a file system hierarchy while only having one copy of the actual data on the disk.

4.2.1 Hard links and symbolic links

The A/UX file system provides two kinds of file links: hard and symbolic. **Hard links** work only within a file system; **symbolic links** can be used within or between file systems. Both kinds of links can point to files or directories. Ordinary users can make symbolic links to either files or directories, but only `root` can make a hard link to a directory.

Files and directories sharing a hard link are actually the same file because they share the same inode, even though the name of the linked file or directory may differ within the file system hierarchy. A symbolic link is actually a special kind of file (with a unique inode) pointing to the inode of the real file or directory.

Symbolic links were invented to allow linked files to be used across file systems. A/UX allows for eight levels of symbolic links and no limit on hard links.

Note: When using NFS, symbolic links are always resolved on

the local system. This means that when remotely mounted file systems contain symbolic links, the actual directories that are pointed to must be either contained in the local directory hierarchy or remotely mounted from the server.

4.2.2 Creating and removing links

To create a hard link to a file, use the `ln` command

```
ln file link
```

For example, assuming `file1` already exists, the commands

```
ln file1 file2
ls -li
```

print something like

```
254 rw-rw-rw- 2 joe doc 29 Mar 14 14:35 file1
254 rw-rw-rw- 2 joe doc 29 Mar 14 14:35 file2
```

where `254` is the inode number, `2` is the number of hard links to this inode, and `29` is the number of bytes in the file.

The `ln` command created `file2`, which shares the same inode (254) with `file1`. Hence, modifying `file2` is the same as modifying `file1`.

To create a hard link to a directory (`root` only), enter

```
ln -f directory link
```

Hard-linked files and directories are removed in the normal way, using `rm` and `rmdir`.

To create a symbolic link to a file or directory, enter

```
ln -s file link
```

or

```
ln -s directory link
```

Assuming directory `dir1` exists, the commands

```
ln -s dir1 dir2
ls -ldi dir1 dir2
```

print something like

```
 25 drwxrwxr-x  2 joe doc 32 Mar 14 14:35 dir1
427 lrwxrwxrwx  1 joe doc  4 Mar 14 14:35 dir2 -> dir1
```

A symbolic link looks and behaves exactly like a normal file or directory except that

- Creating and removing files in a symbolically linked directory actually creates and removes the files in the "real" directory.
- Changing ownership, group, or mode has no meaning (or effect) on a symbolic link. (These must be changed on the "real" files and directories.)

To remove a symbolic link (whether a file or a directory), use the `rm` command. Modifying a symbolic link modifies the "real" file or directory. Removing a symbolic link removes only the pointer to the file, directory, or other symbolic link.

> *Note:* Symbolic links allow up to eight levels of indirection. If you use them frequently, they can turn file systems into a maze. For example, it may be difficult to determine a file's real location if some data is lost and you must restore the data from a backup.

4.2.3 Using NFS and symbolically linked directories

The `/usr` file system contains a number of directories (for example, `/usr/adm`, `/usr/lib/cron`, `/usr/preserve`, `/usr/mail`, `/usr/spool`, and `/usr/tmp`) that administrative functions, time functions, print spoolers, and so on, expect to reside on the local system.

You can share `/usr` over the network and still allow these programs access the local information they expect by creating "private" directories (for example, below `/private`) on the local system and symbolically linking them to the `/usr` directories. For example, the following sequence of commands creates the local directories in

/private:

1. Create the /private directory:

```
mkdir /private
chown bin /private; chgrp bin /private
chmod 755 /private
```

2. Create the usr directory below it:

```
mkdir /private/usr
chown bin /private/usr; chgrp bin /private/usr
chmod 755 /private/usr
```

3. Create /private/usr/adm, owned by adm:

```
mkdir /private/usr/adm
chown adm /private/usr/adm
chgrp adm /private/usr/adm
chmod 755 /private/usr/adm
```

4. Create /private/usr/lib with the owner and permissions as shown:

```
mkdir /private/usr/lib
chown bin /private/usr/lib
chgrp bin /private/usr/lib
chmod 755 /private/usr/lib
```

5. Create "private" directories for cron, preserve, spool, tmp, mail, and any other directories you need in your local /usr:

```
mkdir /private/usr/lib/cron
chown bin /private/usr/lib/cron
chgrp bin /private/usr/lib/cron
chmod 700 /private/usr/lib/cron

mkdir /private/usr/preserve
chown bin /private/usr/preserve
chgrp bin /private/usr/preserve
chmod 755 /private/usr/preserve
```

```
mkdir /private/usr/spool
chown bin /private/usr/spool
chgrp bin /private/usr/spool
chmod 755 /private/usr/spool

mkdir /private/usr/tmp
chown bin /private/usr/tmp
chgrp bin /private/usr/tmp
chmod 777 /private/usr/tmp

mkdir /private/usr/mail
chown bin /private/usr/mail
chgrp bin /private/usr/mail
chmod 777 /private/usr/mail
```

6. After creating these "private" directories, move the contents of the original directories into the private directories, and create symbolic links to their original location. For example, run a shell script such as the following:

```
for i in /usr/adm /usr/lib/cron /usr/preserve \
/usr/spool /usr/tmp /usr/mail
do
    (cd $i ; find . '
-print | cpio -pduvm /private/$i)
    rm -rf $i
    ln -s /private/$i $i
done
```

This allows systems to share a remotely mounted `/usr` while functions that you wish to keep local to each system (for example, administrative functions, time functions, print spoolers, temporary directories, and so on) are accessed through the symbolic link.

Note: If the directory on which a file system is to be mounted is a symbolic link, the file system is mounted on the directory to which the symbolic link refers, rather than being mounted on top of the symbolic link itself.

5. Yellow Pages administration

This section discusses how to administer your system using the Yellow Pages. Topics include how to update and propagate the Yellow Pages maps, how to add a Yellow Pages slave server, and how to use a new master server.

5.1 Server system performance

If you have a single Yellow Pages server on the network and all other systems are Yellow Pages clients, performance on the server will degrade and processes will queue up waiting for an answer on their UID and GID verifications. This will affect every machine on the network. Slow performance can result in timeouts. For example, the server may be too slow to confirm a user name and password before the `login` program times out. You can improve overall performance of the Yellow Pages service by designing the network to include more Yellow Pages slave servers.

5.2 Updating the Yellow Pages maps

You use `make` to update the Yellow Pages maps. After updating the Yellow Pages maps, you use `ypxfr` to propagate the Yellow Pages maps.

5.2.1 Using `make` on the default Yellow Pages maps

The maps derived from the default Yellow Pages files in `/etc` can be easily changed by editing the ASCII files in `/etc` and running `make` from `/etc/yp` on the master server. (See "Checking the Default Files in `/etc`" in Chapter 4.) This will remake any maps that have a more recent ASCII version and propagate copies of any changed maps. This is clearly the easiest method of updating the Yellow Pages.

After making any needed changes to the files, enter

```
cd /etc/yp
make
```

With no arguments, `make` creates `dbm` databases for everything that is out of date, and then executes `yppush` to notify the master server that there has been a change.

You can use a specific map as an argument to `make` (for example, `make passwd`), in which case only the new `passwd` map will be

created and propagated to the slave Yellow Pages servers.

5.3 Propagation of a Yellow Pages map

Propagating a map means copying it from the master Yellow Pages server to slave Yellow Pages server(s). Initially, `ypinit`(1M) copies the maps, as described in "Adding a Slave Server" in this chapter. After a slave server has been initialized, updated maps are transferred from the master server by running `ypxfr`(1M) on the slave machine. `ypxfr` can be run in three different ways:

- periodically by `cron`(1M) on a slave machine
- by the `ypserv`(1M) daemon on a slave machine
- interactively by a user

5.3.1 Periodically by `cron`

Maps have differing rates of change. For example, `protocols.byname` may not change for months, whereas `passwd.byname` may change several times a day in a large organization. A/UX provides sample shell scripts in `/etc/yp` for running `ypxfr` on the various database files: `ypxfr_1h` (once per hour), `ypxfr_1d` (once per day), and `ypxfr_2d` (twice per day). These scripts should be modified for the needs of your site and then entered in the appropriate `crontab`(4) file of each slave server in the domain.

Stagger the execution time to avoid slowing down the master server. Should you want a map transferred from a server other than the master, specify the `-h` flag option to `ypxfr` in the appropriate shell script. You may also use a `crontab` entry to invoke `ypfxr` for maps with unique update requirements.

5.3.2 By `ypserv`

A master server can request that all slave servers within a domain update certain maps. This update request is initiated by the `yppush`(4) command issued from the master machine. `yppush` sends a "transfer map" request on the network, and the `ypserv` daemon on each slave responds by executing `ypxfr` to update the requested map. The next subsection gives typical command lines for `yppush` and `ypxfr`.

5.3.3 Interactively by a user

Typically, `ypxfr` is run interactively only in exceptional situations. Examples are setting up a temporary Yellow Pages server to create a test environment, or trying to get a Yellow Pages server that has been out of service quickly consistent with the other servers. A typical command line is

```
/etc/yp/ypxfr map-name
```

where *map-name* is a recognized name or nickname for a Yellow Pages map.

Usually, you use `yppush` to invoke `ypxfr` interactively. `yppush` invokes `ypxfr` on each slave machine to update a particular Yellow Pages map. A typical use would be updating the `passwd.byname` map on all slaves immediately after using `yppasswd` to change a user password. A typical command line is

```
/etc/yppush map-name
```

where *map-name* is a recognized name or nickname for a Yellow Pages map.

See Tables 4-1 and 4-2 for lists of valid map names to use with `ypxfr` and `yppush`.

5.4 Adding a slave server

This section assumes that the slave server machine is already on your network and running the appropriate network and NFS daemons. Adding a new slave server to a domain involves two basic steps:

1. On the master server, modify the appropriate host and server databases to include the new slave machine and then propagate the updated databases to the existing slave machines.
2. On the new slave server, use `ypinit` to initialize the Yellow Pages databases from the master machine.

5.4.1 Modifying the master server's databases

To add the new slave host name to the `ypsrvs` database,

1. Log in as `root` on the master server and enter the commands

```
cd /etc/yp
makedbm -u 'domainname'/ypsrvs > ypsrvs.tmp
```

Note: All Yellow Pages databases are in `dbm`(3X) format. This step makes an ASCII version of the `ypsrv` database in `ypsrvs.tmp`. For more information, see `dbm`(3X) and `makedbm`(1M) in *A/UX Programmer's Reference*.

2. Edit `ypsrvs.tmp`. Initially, your file will look something like this:

```
YP_LAST_MODIFIED 0565942409
YP_MASTER_NAME hostname1
hostname5
hostname6
```

   Add a line to the end of the file containing the new slave host name.

3. Enter the commands

```
makedbm ypsrvs.tmp tmpmap
mv tmpmap.dir 'domainname'/ypsrvs.dir
mv tmpmap.pag 'domainname'/ypsrvs.pag
```

   These commands remake the `ypsrvs` database by recreating `ypsrvs.pag` and `ypsrvs.dir`.

4. Check your work. Enter the command

```
makedbm -u 'domainname'/ypsrvs
```

   This command should show you a list of all the Yellow Pages servers. If it does not, redo all the steps in this immediate subsection, beginning with step 1 above.

5. When you have ensured that the list of Yellow Pages servers is correct, remove the temporary file with the command

```
rm ypsrvs.tmp
```

If `/etc/hosts` does not contain an entry for the new slave server, make one. If the machine is already on the network, the entry should be there and *you can skip the remaining steps of this section.*

To update the host `hst.nm` and `hst.ad` databases to contain the new slave host name and address:

1. Use a text editor to make an entry for the new slave server in `/etc/hosts`.
2. Enter these commands:

   ```
   cd /etc/yp
   rm hosts.time
   make hosts
   ```

   The Makefile in `/etc/yp` specifies that when `make` is run without a `NOPUSH=1` argument, `yppush`(1) will automatically be run to force database updates on the existing slave machines. Removing `hosts.time` ensures that `make` will make a new hosts database.

5.4.2 Initializing databases on the slave server

To initialize the database on the new slaver server:

1. Log in as `root` on the new slave server. Check the second field of `/etc/HOSTNAME` for the domain name. It must agree with the master machine's domain name. If necessary, use a text editor to change this field to the correct domain name (tabs or blanks are the field separators), and reboot A/UX.
2. Enter

   `/etc/yp/ypinit -s` *master-name*

   where *master-name* is the name of the master server. The `ypinit` command retrieves the Yellow Pages databases from the master server.
3. Change the third field in `/etc/inittab` for the `/etc/ypbind` and `/etc/ypserv` entries from `off` to `wait`.

4. Start the `ypbind` and `ypserv` daemons from the terminal, or simply reboot the slave.

5.5 Using a new master server

When changing master servers, follow one of the two procedures in this section: follow the first when the master is out of service; follow the second when the master is running and you want to convert it to a slave server.

5.5.1 Old master out of service

The first problem in creating a new master when the old master is not online is getting the slave servers to believe the new master is actually the new master. This conversion process is inherently an inefficient one because you must make changes to the Yellow Pages database on each slave machine in the old master's domain.

The simplest approach is to convert one of the existing slave machines to the master. The following is a reasonable solution:

1. Log in as `root` on the slave that you are converting to a master. Use `ps`(1) to determine the process IDs of `/etc/ypserv` and `/etc/ypbind`. Use the `kill`(1) command to terminate these processes. Killing the Yellow Pages daemons ensures that your work will not be undone by normal Yellow Pages update processes (`ypxfr` and `yppush`).
2. Enter the commands

```
cd /etc/yp
makedbm -u `domainname`/ypsrvs > ypsrvs.tmp
```

   Note: All Yellow Pages databases are in `dbm`(3X) format. This step makes an ASCII version of the `ypsrvs` database in `ypsrvs.tmp`. For more information, see `dbm`(3X) and `makedbm`(1M) in *A/UX Programmer's Reference*.

3. Edit `ypsrvs.tmp`. Initially, your file will look something like this:

```
YP_LAST_MODIFIED 0565942409
YP_MASTER_NAME old-master
hostname5
hostname6
old-slave
```

Delete the line containing the old slave host name. Change the master's host name in the field following `YP_MASTER_NAME` to the new master's host name. (The field separator is a blank or tab character.) Save the file.

4. Enter the commands

```
makedbm ypsrvs.tmp tmpmap
mv tmpmap.dir 'domainname'/ypsrvs.dir
mv tmpmap.pag 'domainname'/ypsrvs.pag
```

These commands remake the `ypsrvs` database by recreating `ypsrvs.pag` and `ypsrvs.dir`.

5. Check your work. Enter the command

```
makedbm -u 'domainname'/ypsrvs
```

This command should show you a list of all the Yellow Pages servers. If it does not, redo all the steps in this immediate subsection, beginning with step 1 above.

6. When you have ensured that the list of Yellow Pages servers is correct, remove the temporary file with the command

```
rm ypsrvs.tmp
```

7. On the remaining slave hosts, overwrite `/etc/yp/'domainname'/ypsrvs.page` and `/etc/yp/'domainname'/ypsrvs.dir` with the new version you created on this, the new master host. The `ftp` program as described in "Adding an A/UX System to the Network" in Chapter 5 is a reasonably efficient method for doing this.

8. Restart the Yellow Pages daemon. Enter

```
/etc/ypserv
/etc/ypbind
```

9. Check the Yellow Pages daemon entries in `/etc/inittab` to be sure that the third (colon separated) fields are set to `wait`. (This insures that the Yellow Pages daemons are started every time your machine enters multi-user mode.)

5.5.2 Old master in service

It is easier to change from one master server to another when the old master is still up and running. The trick is to convert the old master to a slave by changing the `ypsrvs` database to a slave database. Since the slave servers believe the old master is still the master server, invoking `yppush` on this machine will propagate the change to the slaves. The slave servers will then respond to the new master server.

1. Log in as `root` on the old master that you are converting to a slave.
2. Enter the commands

```
cd /etc/yp
makedbm -u `domainname`/ypsrvs ypsrvs.tmp
```

Note: All Yellow Pages databases are in `dbm`(3X) format. This step makes an ASCII version of the `ypsrvs` database in `ypsrvs.tmp`. For more information, see `dbm`(3X) and `makedbm`(1M) in *A/UX Programmer's Reference*.

3. Edit `ypsrvs.tmp`. Change the master's host name in the field following `YP_MASTER_NAME` to the host name of the new master server. Add the old master's name on a line at the end of the file. When you are finished, your file will look something like this:

```
YP_LAST_MODIFIED 0565942409
YP_MASTER_NAME new-master
hostname5
hostname6
old-master
```

4. Enter the commands

```
makedbm ypsrvs.tmp tmpmap
mv tmpmap.dir 'domainname'/ypsrvs.dir
mv tmpmap.pag 'domainname'/ypsrvs.pag
```

   These commands remake the `ypsrvs` database by recreating `ypsrvs.pag` and `ypsrvs.dir`.

5. Check your work. Enter the command

```
makedbm -u 'domainname'/ypsrvs
```

   This command should show you a list of all the Yellow Pages servers. If it does not, redo all the steps in this immediate subsection, beginning with step 1 above.

6. When you have ensured that the list of Yellow Pages servers is correct, remove the temporary file with the command

```
rm ypsrvs.tmp
```

7. Distribute the data to the other slave servers with the command

```
/etc/yp/yppush ypsrvs
```

   The old master is now running as a slave server, and the other slave servers should respond to the new master. You may now do what you will with the old master.

6. A network mail system

The `sendmail` program is not a user-interface routine; it simply delivers preformatted messages. The `sendmail` program is invoked as a daemon from `/etc/inittab` as described in the following subsection. See `sendmail`(1M) for a description of all the flag options available.

The `sendmail` program lets you mail messages across a network. If you address mail to a user without specifying a host name, mail is delivered to the local user. If you address mail to *site!user* or *user@site.uucp*, the mail message is routed via UUCP. If a direct connection does not exist, then the mail is forwarded to the relay `L_RELAYHOST`. If you address mail to *user@site*, the mail message is sent via TCP/IP if possible. If the site is not connected via TCP/IP, the mail message is delivered via UUCP if the site is a directly connected UUCP host; otherwise, the message is forwarded to the relay.

You must specify a complete route for addresses on networks that are not directly connected to your network or use a smart router. For example, if you want to reach a user `zippy` on host `pinhead` that is connected to your host through host `comix`, you must address your UUCP message as either

```
comix!pinhead!zippy
```

or

```
comix!"zippy@pinhead"
```

The `sendmail` configuration file, `/usr/lib/sendmail.cf`, handles domain names correctly, both Yellow Pages and Internet, in the current version of A/UX.

6.1 Starting the `sendmail` daemon

The `sendmail` daemon is invoked in `/etc/inittab`. This daemon performs two functions: It listens on the SMTP (Simple Mail Transfer Protocol) socket for connections (to receive mail from a remote system), and it processes the queue periodically to ensure that mail gets delivered when hosts come up.

You can start the `sendmail` daemon immediately by following steps 4 and 5 below, or you can first customize the `sendmail` configuration as outlined in the following steps.

1. To customize your `sendmail` configuration, edit these fields in the file `/usr/lib/sendmail.m4`:

   - `L_SITE_M` specifies the primary host name of your machine. If your domain server is set up to return simple

one token names, you do not need to modify this field.

- ☐ `L_SITE_C` specifies all the alias host names by which your machine is known.
- ☐ `L_DOMAIN_M` specifies the main domain name that your machine resides in. Do not define this field if you communicate only via UUCP.
- ☐ `L_DOMAIN_C` specifies the list of aliases by which this domain is also known.
- ☐ `L_UNIVERSE_M` specifies the top-level domain in which your machine resides. Examples of top-level domains are gov, edu, and com.
- ☐ `L_UNIVERSE_C` specifies other top level domains your machine is in.
- ☐ `L_USERELAY` is defined if and only if there is some machine which should receive mail directed to unknown hosts.
- ☐ `L_RELAYHOST` specifies the name of the host to which mail to unknown machines should be redirected.
- ☐ `L_RELAYMAILER` specifies the mailer to use to get mail to the relay.
- ☐ `L_HIDEHOST` is defined if mail from this host should appear to be from some other host. In this case, define this field to be the name of the host from which the mail should appear to come. Note that this works only if `L_HIDEHOST` is connected via TCP/IP.

2. Install the customized configuration

```
m4 /usr/lib/sendmail.m4 > /usr/lib/sendmail.cf
```

3. Freeze the newly installed configuration

```
/usr/lib/sendmail -bz
```

4. Start the `sendmail` daemon by editing `/etc/inittab` to change the status field (shown in bold) of the following line from

```
net8:2:off:/usr/lib/sendmail -bd -q30m
```

to

```
net8:2:once:/usr/lib/sendmail -bd -q30m
```

This line uses two flags:

`-bd` Causes `sendmail` to listen on the SMTP port.

`-q30m` Causes `sendmail` to run the queue every half hour.

5. Reboot A/UX.

6.2 `sendmail` Implementation notes

Domain servers are supported in this version of `sendmail`. `MXDOMAINS` are turned on, but as implemented by the University of California at Berkeley. This means that only `T_MX` and `T_CNAME` records are really used. Also, site canonicalization is done by asking for all records matching a given name, thus `T_MB` records and similar records can confuse the issue. The only real problem is that an invalid address that looks like *user@site* could be considered valid if *site* corresponds to a user with a `T_MB` record.

Chapter 8
Tools for Checking System Status

Contents

1. Introduction 1

2. Determining network status 2
   2.1 What hosts are up and who is logged in 2
   2.2 Determining if a host is up 3
   2.3 Debugging network problems 5

3. Determining NFS status 9
   3.1 Identifying which file systems are mounted 10
   3.2 Identifying which clients have mounted systems 10
   3.3 Determining the status of NFS servers and clients 11
   3.4 Debugging Yellow Pages and `mount` problems 11

4. Determining Yellow Pages status 13
   4.1 The domain name 13
   4.2 Determining if Yellow Pages maps have been propagated 14
   4.3 Identifying the server 14
   4.4 Examining Yellow Pages maps 14

Chapter 8

Tools for Checking System Status

1. Introduction

There are several tools available for determining various parameters of the network. Many of these utilities are located in the `/etc` or `/usr/etc` directories. If these directories are not included in the value of the `PATH` variable in `root` `.login` or `.profile` files, add them to the `PATH` value.

The following utilities are particularly useful in network system administration:

`ruptime`
: Show host status of local machines. See `ruptime`(1N).

`rwho`
: Show who is logged in on local machines. See `rwho`(1N).

`ping`
: Send ICMP (Internet Control Message Protocol) `ECHO_REQUEST` packets to network hosts. See `ping`(1M).

`netstat`
: Show network status. See `netstat`(1N).

`df`
: Report the mounted file systems and the number of disk blocks. Note that the number of free inodes shows as zero on remote file systems. Also note that this command will hang the client process if a remote file server is down. See `df`(1).

`mount`
: Show mounted file systems. Note that this command will not hang the client process if the remote file server is down, whereas the `df` command will. See `mount`(1M).

`showmount`
: Show all remote mounts. See `showmount`(1M).

`nfsstat`
: Display statistical information about the network file systems. See `nfsstat`(1M).

`rpcinfo`
: Report RPC (remote procedure call) information. See `rpcinfo`(1M).

`domainname`
: Without an argument, display the Yellow Pages domain name for the current system. See `domainname`(1).

`ypcat`
: Print the values in a Yellow Pages database. See `ypcat`(1).

`ypmatch`
: Print the value of one or more keys from a Yellow Pages map. See `ypmatch`(1).

`yppoll`
: Display what version of a Yellow Pages map is at a Yellow Pages server host. See `yppoll`(1M).

`ypset`
: Change a map's current Yellow Pages server. See `ypset`(1M).

`ypwhich`
: Display which machine is the Yellow Pages server. See `ypwhich`(1M).

This chapter surveys the use of these commands. See the appropriate manual page entry for full information.

2. Determining network status

You can use a number of commands to determine the status of your network. This section discusses the `ruptime`, `rwho`, `ping`, and `netstat` commands.

2.1 What hosts are up and who is logged in

The `ruptime` and `rwho` commands are used to determine what hosts are up and who is logged into those hosts. However, they rely on the `rwhod` daemon, which periodically broadcasts and updates its

status list. This daemon causes a lot of overhead if the network is large and if different types of systems on the network accept broadcasts differently, and it may be turned off on one or more systems on the network. For example, suppose you enter

```
ruptime; rwho
```

on `hostname1` and see the response

```
hostname7   down 58+17:09
hostname4   down 55+15:18
hostname3   down 52+17:14
hostname6   down 52+17:01
hostname1     up     0:11
hostname5   down 52+18:04
hostname2   down 52+02:42

holly    hostname1:console Oct 6 18:31
```

This response means either that your machine is the only machine up and running on the network or that your machine is the only one running `rwhod`. If `ruptime` prints

```
no hosts!?!
```

you know that the local host is not running `rwhod`.

2.2 Determining If a host Is up

The `ping` command is the simplest way of determining whether a host is really up. It uses ICMP's mandatory `ECHO_REQUEST` datagram to elicit a response from a host or gateway.

If `hostname1` is not responding, for example, to an NFS `mount` request, enter

```
ping hostname1
```

from the client machine. This should print something similar to

```
64 bytes from 128.8.1.1: icmp_seq=0. time=16. ms
64 bytes from 128.8.1.1:  icmp_seq=1. time=16. ms

(interrupt)

----hostname1 PING Statistics----
2 packets transmitted, 2 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 16/16/16
```

if the network and both systems are functioning. You can use the interrupt character (usually CONTROL-s) to cause `ping` to exit and print statistics. The `ping` command prints

```
no answer from hostname1
```

if `hostname1` is down, or

```
ping: Network is unreachable
```

if the network is not functioning. (In this case, check all Ethernet connections.)

When using `ping` to find out why systems are not communicating properly, run it first on the local system. On `hostname1` enter

```
ping hostname1
```

If this prints something similar to

```
64 bytes from 128.8.1.1: icmp_seq=0. time=16. ms
64 bytes from 128.8.1.1:  icmp_seq=1. time=16. ms

(interrupt)
----hostname1 PING Statistics----
2 packets transmitted, 2 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 16/16/16
```

the local network interface is up and running. You can then use `ping` on other systems that are farther and farther away, through forwarder systems, and so on.

2.3 Debugging network problems

The `netstat` command is useful in examining the contents of various network-related data structures for a local network running BSD-derived networking software, such as B-NET.

For statistics on active interfaces, enter

```
netstat -i
```

The response should be something like

```
Name Mtu  Network     Address    Ipkts Ierrs Opkts Oerrs Collis
ae0  1500 128.8       hostname1 40506 0     10158 0     0
lo0  1536 loopback-n  loop      1012  0     1012  0     0
```

giving the number of input packets, output packets, input errors, output errors, and collisions, respectively. With an "interval" argument, `netstat -i` displays a running count of statistics related to network interfaces.

For example,

```
netstat -i 5
```

prints the statistics detailed above, as well as incremental data every 5 seconds. The first line of each screen of information contains a summary since the system was last rebooted. Subsequent lines of output show values accumulated over the preceding interval.

```
netstat -n
```

prints the active Internet connections by using Internet addresses rather than host names.

For statistics on active routes, enter

```
netstat -r
```

This prints something like

```
Routing tables
Destination Gateway         Flags   Refcnt Use    Interface
loop        loop            UH      0      0      lo0
128.8.200   hostname2       UG      0      2228   ae0
128.8.100   hostname1       U       13     1337   ae0
```

(This will produce a big table on a network with a lot of Internet routers and gateways.)

For information about listening sockets and active connections, enter

```
netstat -a
```

The response is something like

```
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address  Foreign Address (state)
tcp        0      0  hostname1.111  hostname1.1185  CLOSE_WAIT
??? new connections created
tcp        0      0  hostname1.1185 hostname1.111   TIME_WAIT
tcp        0      0  hostname1.1184 hostname1.111   TIME_WAIT
tcp        0      0  hostname1.1183 hostname1.111   TIME_WAIT
tcp        0      0  *.smtp         *.*             LISTEN
tcp        0      0  *.ftp          *.*             LISTEN
tcp        0      0  *.telnet       *.*             LISTEN
tcp        0      0  *.shell        *.*             LISTEN
tcp        0      0  *.login        *.*             LISTEN
tcp        0      0  *.exec         *.*             LISTEN
tcp        0      0  *.finger                       LISTEN
tcp        0      0  *.111          *.*             LISTEN
udp        0      0  *.1018         *.*
udp        0      0  *.1019         *.*
udp        0      0  *.1020         *.*
udp        0      0  *.1021         *.*
udp        0      0  *.1022         *.*
udp        0      0  *.1023         *.*
udp        0      0  *.tftp         *.*
udp        0      0  *.talk         *.*
udp        0      0  *.1037         *.*
udp        0      0  *.1035         *.*
udp        0      0  *.1033         *.*
udp        0      0  *.biff         *.*
udp        0      0  *.route        *.*
udp        0      0  *.111          *.*
```

Finally, to see statistics on the various protocols, enter

```
netstat -s
```

The response should be something like

```
udp:
   0 incomplete headers
   0 bad data length fields
   0 bad checksums
tcp:
       1989 packets sent
              614 data packets (19177 bytes)
              4 data packets (1229 bytes) retransmitted
              1288 ack-only packets (145 delayed)
              3 URG only packets
              9 window probe packets
              0 window update packets
              71 control packets
       2148 packets received
              677 acks (for 19257 bytes)
              46 duplicate acks
              0 acks for unsent data
              565 packets (4520 bytes) received in-sequence
              1047 completely duplicate packets (1047 bytes)
              0 packets with some dup. data (0 bytes duped)
              25 out-of-order packets (0 bytes)
              0 packets (0 bytes) of data after window
              0 window probes
              1 window update packet
              0 packets received after close
              0 discarded for bad checksums
              0 discarded for bad header offset fields
              0 discarded because packet too short
       24 connection requests
       25 connection accepts
       49 connections established (including accepts)
       47 connections closed (including 1 drop)
       0 embryonic connections dropped
       672 segments updated rtt (of 701 attempts)
       4 retransmit timeouts
              0 connections dropped by rexmit timeout
       0 persist timeouts
       0 keepalive timeouts
```

```
            0 keepalive probes sent
            0 connections dropped by keepalive
icmp:
   165 calls to icmp_error
   0 errors not generated 'cuz old message was icmp
   Output histogram:
      echo reply: 1
      destination unreachable: 165
      information request reply: 86
   0 messages with bad code fields
   0 messages < minimum length
   0 bad checksums
   0 messages with bad length
   Input histogram:
      echo reply: 3
      echo: 1
   87 message responses generated
ip:
   16020 total packets received
   0 bad header checksums
   0 with size smaller than minimum
   0 with data size < data length
   0 with header length < data size
   0 with data length < header length
   0 fragments received
   0 fragments dropped (dup or out of space)
   0 fragments dropped after timeout
   0 packets forwarded
   0 packets not forwardable
   0 redirects sent
```

3. Determining NFS status

This section describes commands useful in administering NFS. Check the manual page entry for each of these commands for an explanation of all the possible options.

3.1 Identifying which file systems are mounted

The `df` command prints the mounted file systems and the number of free disk blocks on each file system. The `df` command reports zero inodes free on the remote file systems. It is not necessarily true that there are no inodes free. The current NFS protocol specification does not include a way to obtain the number of free inodes on the remote system, so the number always appears as zero on remotely mounted file systems. On the client machine, `df` without an argument displays something like

```
/      /dev/dsk/c0d0s0      15542 blocks   2358 inodes
/h1    hostname1:/users     22828 blocks      0 inodes
```

The `mount` command without an argument simply prints the mounted file systems and will not hang if the server is down. On the client machine, `mount` without an argument prints something like

```
/dev/dsk/c0d0s0 on  /   type 5.2 (rw)
hostname1:/users on /h1 type nfs (rw,soft,noquota)
```

3.2 Identifying which clients have mounted systems

On the server machine, the command

```
showmount
```

lists the clients that have remotely mounted a file system from the current system:

```
hostname2
```

The command

```
showmount -e
```

on the server machine lists file systems that are currently exported:

```
/              hostname2
```

On the client machine, the command

```
showmount -e hostname1
```

lists file systems exported by `hostname1`:

```
export list for hostname1:
/      hostname2
```

3.3 Determining the status of NFS servers and clients

The `nfsstat` command displays or reinitializes statistical information about NFS and RPC. This is useful in troubleshooting NFS service and the network. For example, to print NFS client information, enter

```
nfsstat -cn
```

The response should be something like

```
Client nfs:
calls    badcalls  nclget   nclsleep
7644     0         7644     0
null     getattr   setattr  root     lookup   readlink  read
0  0%    225  2%   6  0%    0  0%    992 12%  0  0%     3408 44%
wrcache  write     create   remove   rename   link      symlink
0  0%    2772 36%  60  0%   1  0%    0  0%    0  0%     0  0%
mkdir    rmdir     readdir  fsstat
0  0%    0  0%    177  2%   3  0%
```

3.4 Debugging Yellow Pages and `mount` problems

Once you have established that a host is up, you can run `rpcinfo`. The `rpcinfo` command reports information about what RPC programs are being served by the specified host. With the `-p` option, it probes the portmapper and lists all the RPC programs running. For example, if you are on `hostname2` and enter

```
rpcinfo -p hostname1
```

the response should be

```
program   vers   proto   port
 100007     2      udp   1027   ypserv
 100007     2      tcp   1030   ypserv
 100007     2      udp   1027   ypserv
 100007     2      tcp   1030   ypserv
 100007     2      tcp   1031   ypbind
 100007     2      udp   1035   ypbind
 100007     2      tcp   1031   ypbind
 100007     2      udp   1035   ypbind
 100003     2      udp   2049   nfs
 100012     1      udp   1062   sprayd
 100008     1      udp   1064   walld
 100005     1      udp   1066   mountd
 100002     1      udp   1068   rusersd
 100002     2      udp   1068   rusersd
 100001     1      udp   1071   rstatd
 100001     2      udp   1071   rstatd
```

If a remote host is not specified, `rpcinfo` lists the RPC programs registered on the local system. A file called `/etc/rpc` (similar in form to `/etc/services`) lists some of the program numbers and the programs associated with them; for example,

```
rstatd          100001  rstat rup perfmeter
rusersd         100002  rusers
nfs             100003  nfsprog
ypserv          100004  ypprog
mountd          100005  mount showmount
ypbind          100007
walld           100008  rwall shutdown
yppasswdd       100009  yppasswd
etherstatd      100010  etherstat
rquotad         100011  rquotaprog quota rquota
sprayd          100012  spray
selection_svc   100015  selnsvc
```

The port numbers will be different for each system. In this case, the number representing the `rpc.mountd` process is `100005`. (`100004` represents the `ypserv` process, and `100007` represents

the `ypbind` process.)

You can also use

```
rpcinfo -u host-name  nfs 2
```

to see if the NFS server on *host-name* can be reached from the local machine.

If the portmap daemon is not running, `rpcinfo` will fail with the message

```
rpcinfo: can't contact portmapper: RPC_SYSTEM_ERROR -
        Connection refused
```

In this case, you should restart the portmapper and kill any RPC daemons that are running (for example, `rpc.mountd` and `rpc.rstatd`):

```
/etc/portmap
kill -9 rpc.daemon.PID
```

PID is the process number associated with the RPC daemon.

> *Note:* This method relies on `init` respawning the `inetd` daemon after it is killed, which it will do if `/etc/inittab` has been modified as shown in Chapter 2, "Establishing a Two-System Network."

4. Determining Yellow Pages status

This section describes commands useful in administering the Yellow Pages. Check the manual page entry for each of these commands for an explanation of all the possible options.

4.1 The domain name

The command

```
domainname
```

displays the name of the current Yellow Pages domain. This is a useful check if your network uses or has access to more than one domain, or if you are not certain the domain name has been set on a particular

machine. If the domain name on a client or slave server system is not consistent with the domain name on the master server, the Yellow Pages will have serious problems.

4.2 Determining if Yellow Pages maps have been propagated

The `yppoll` program asks any `ypserv` daemon for the information it holds internally about a single map. It is used to find out if a new version of the database has been propagated to a particular host. For example, the command

```
/etc/yp/yppoll -h hostname1 passwd.byname
```

(where `hostname1` is the Yellow Pages master server) returns the information

```
Map passwd.byname has order number 512391909.
The master server is hostname1.
```

4.3 Identifying the server

The binding of Yellow Pages client to Yellow Pages server changes when the network or servers are very busy. Whenever possible, the system stabilizes at a point where all clients get acceptable response time from the Yellow Pages servers. The `ypwhich` command returns the host name of the server machine that is currently being used to access the Yellow Pages. The command

```
ypwhich
```

prints the name of the Yellow Pages server machine. The answer you get back from `ypwhich` will vary as the Yellow Pages server changes. This is normal.

As long as your client machine gets Yellow Pages service, it doesn't matter where the service comes from. Often a Yellow Pages server machine gets its own Yellow Pages services from another Yellow Pages server on the net.

4.4 Examining Yellow Pages maps

The command

```
ypcat passwd
```

prints out the values in a specified Yellow Pages map, in this case the `passwd` map. If no map is specified, it prints a usage message.

The `ypmatch` command prints the values associated with one or more keys from the Yellow Pages map (database) specified by a name, which may be either a map name or a map nickname. For example, `passwd` is a nickname for the map named `passwd.byname`; enter

```
ypmatch -x
```

to see the map nickname table.

The keys you specify must be exact values in their capitalization and length. No pattern matching is available. If a key is not matched, a diagnostic message is produced. For example, the command

```
ypmatch hostname1 hosts
```

returns the Internet address for `hostname1`.

Chapter 9
Troubleshooting

Contents

1. Introduction 1

2. Assessing system problems 2
   2.1 Keeping a record of system activity 2
   2.2 Using a specific description 2
   2.3 Determining whether it is a hardware or software problem 2

3. Analyzing software problems 4
   3.1 Console error messages 4
      3.1.1 A "`filesystem full`" message . . . 5
      3.1.2 NFS or Yellow Pages messages 5
      3.1.3 Other messages 6
   3.2 Network error messages 6
   3.3 Symptoms and tools for analysis 7
      3.3.1 Cannot reach a machine 8
      3.3.2 Cannot log in 9
      3.3.3 A "`Connection timed out`" message 10
      3.3.4 Cannot NFS-mount a file system 11
      3.3.5 Things work, but slowly 11

4. Debugging the network services 11
   4.1 Debugging the network hardware 11
      4.1.1 Software checks 12
      4.1.2 Hardware checks 13
   4.2 Debugging the NFS 14
      4.2.1 Remote mount failure: Hard mount vs. soft mount 14
      4.2.2 Checking the NFS server 15

4.2.3 Checking the NFS client daemons 17
4.3 Debugging the Yellow Pages 17
4.3.1 Checking **ypserv** and **ypbind** 17
4.3.2 Checking the Yellow Pages server 17
4.3.3 Checking the Yellow Pages client 18
4.4 Specific problems in the network environment 19
4.4.1 Remote mount hangs during boot 19
4.4.2 Processes hang 19
4.4.3 Processes time out 20
4.4.4 Things work, but slowly 20
4.4.5 Yellow Pages errors 21
4.4.6 Commands hang on Yellow Pages client 21
4.4.7 Cannot log in 22
4.4.8 Yellow Pages commands terminate with a message 22
4.4.9 Directory listing reports numbers 22
4.4.10 **ypbind** crashes 23
4.4.11 **ypserv** crashes 25
4.5 Error messages in the network environment 25

Chapter 9

Troubleshooting

1. Introduction

This chapter discusses general error conditions for B-NET, NFS, and Yellow Pages and provides recommended solutions. This chapter also describes symptoms of troubled systems and presents strategies for breaking the troubleshooting process into manageable steps. Here are a few basic guidelines:

- Always look for specific symptoms. When you first become aware of a system problem, ask yourself: What happened? When? Where? How? Focusing on the answer to these questions will give you a good start on assessing what's gone wrong.
- Make a note of system maintenance or unusual system behavior that may have occurred near the time of a problem and thus may have somehow contributed to the problem.
- Check for the most simple kind of failure first. Only consider more complex problems when you have eliminated the simple causes of failure.
- Choose the solution that will have the smallest impact on the resources of your machine or your network. (You may be able to avoid rebooting in many cases.)
- If you have to take a machine down temporarily, be aware of what services, if any, it provides to other machines and try to warn users of the halted service.

Troubleshooting falls into three steps: assessment, analysis, and action. The sections that follow are representative of these steps.

2. Assessing system problems

This section gives suggestions on how to assess system problems.

2.1 Keeping a record of system activity

Keep a system log that records maintenance and problems for every machine at your site. This is usually a simple notebook that is accessible to other people if you are unavailable. Don't keep this record on-line; if the system goes down, you won't be able to read it!

Keep an on-line record of any software changes made on each machine. The method will vary from site to site, but the general idea is to have a log file that all users with root privilege can read and write, and that they can look at to see what's been changed recently on the machine. For example, if you install a new version of a program or of some file or database the system uses regularly, you should note that in the on-line log. Likewise, note deletions or changes to existing files.

2.2 Using a specific description

Get as specific a description as possible of the trouble. Find out who the users are, what machines they are working on, what they were doing when the problem occurred, and anything else relevant or unusual.

If you discover the trouble yourself rather than hearing about it second-hand, you can begin narrowing it down by checking system logs for recent changes or maintenance in the affected area.

Find out if other users or machines are having similar problems. If only one machine has a problem, you can safely assume the problem is confined to that machine, on either the hardware or the software level. If several machines or users have a similar problem, you should look for a server or network failure.

2.3 Determining whether it is a hardware or software problem

Try to determine whether the trouble is related to hardware or software. Sometimes a hardware problem generates the same symptoms as a software problem; this is one area where experience really helps in making diagnoses.

However, there are a few symptoms that almost always indicate hardware problems:

- Problems that can be described as "the system is dead" are most often related to hardware. It can mean there's no power or no connection whatsoever. When screens or monitors are blank or dark, you may have a power failure within the machine, or a bad connection because of a loose plug, a loose or blown fuse, or even a machine that isn't switched on. Check all of these to see if you need to replace parts.
- If a machine suddenly and permanently loses a network connection, there could be a problem with network cabling. (Note *permanently* here, because there are software problems that cause degradation of network service.) Has there been any work at your site on the network itself? In general, network hardware problems are harder to pinpoint than other hardware problems. When network hardware fails, it doesn't generate error messages. However, a broken network will interrupt NFS or Yellow Pages service, and you will see an error message from one or both of those services. As a general rule you should treat NFS or Yellow Pages error messages as if there has been a failure in one of those services. See the appropriate sections in this chapter. If you cannot find a problem with the services or their database files, you should look for network hardware problems. See "Debugging the Network Hardware" in this chapter.
- Other kinds of hardware problems sometimes show up when machines fail to boot properly, or crash and then fail to boot, with messages that include the words `memory error`. Such messages are rare and usually indicate a bad memory board.
- Occasionally, on the machine console you will see a message that includes the word `panic` followed by an error message. Panic messages mean that the system has crashed. This sometimes indicates a hardware problem. Copy the error message exactly and save the copy. Attempt to reboot the system if it has not already attempted to reboot on its own. Failure to reboot strongly suggests a hardware problem.

- Another message sometimes seen on the console is of the form

  `err on dev` *name*

  where *name* is the device name for a particular disk. When the system tries to read a particular disk block and cannot do so, it prints this message. The system then tries to reread the block. Disk errors of this type are common and may be ignored unless they occur in large quantities—many dozens. When there are many such messages, you probably have a bad spot on the disk. It has to be mapped out with the `autorecovery`(8) program.

This list doesn't cover all possibilities. You may, over time, add more kinds of hardware failure to it. Hardware problems are relatively rare, but it's good to know a bit about the symptoms. When there is a hardware failure, you will probably have to call a hardware technician or a maintenance contract service. These arrangements will vary from site to site and cannot be specifically described here. Know what yours are, and keep necessary phone numbers and names posted in known places.

It's much more common to have a problem with system software. The remainder of this chapter deals mainly with ways to analyze software problems and actions to take to fix them.

3. Analyzing software problems

There are several kinds of system failure or poor system performance that can be caused by what's running on the system, and not by the hardware components of the system. If you have determined as best you can that your system's hardware is in good working order, you need to look further for the problem. This section guides your search. First it analyzes some common error messages that typically indicate software problems, and suggests corrective action. Next it presents symptoms you might encounter and introduces some software tools you might use to narrow down the type of problem or its location.

3.1 Console error messages

This section contains messages that are usually generated by problems that can be fixed through software alteration; these do not indicate hardware problems. However, a message such as "`not`

`responding`'' may indicate either a software problem or a loose connection, and it makes sense to always check your network connections first. On systems running a window package, these messages appear on the console, disrupting the windows on the screen.

3.1.1 A "`filesystem full`" message

If a particular file system runs out of space, the system will print messages on the console of the form

```
fserr: filesystem full
no space on dev name
```

where *name* is the device name for a file system on your system. If the message recurs regularly, you must make space on the device indicated. Usually this means removing unneeded files. Check carefully before removing anything. Make a tape backup of files to be removed if there is *any* possibility they will be needed in the future.

3.1.2 NFS or Yellow Pages messages

Occasionally, there will be messages of the form

```
yp: server not responding for domain domain-name
```

or

```
NFS: server s not responding; still trying...
```

where *s* is the server's name.

Usually this second one will be followed shortly by the message

```
NFS server s server OK.
```

In such cases, you may ignore the message; it was probably generated in response to slow communication over a heavily loaded network. However, on rare occasions there will be continuous ''`server not responding`'' messages for several minutes. You should then check the server machine and the network traffic load to make sure nothing is broken. See ''Symptoms and Tools for Analysis'' and ''Debugging the Yellow Pages,'' both later in this chapter. Finally, check the network hardware if you haven't turned up any abnormalities. See ''Debugging the Network Hardware'' later in this chapter.

3.1.3 Other messages

The message

```
stale NFS file handle
```

most often occurs on a NFS client system after the NFS server has been rebooted. If you see this message, unmount the remote file system, kill the `rpc.mountd` process, then remount the remote file system.

The system also produces other messages, such as

```
Random interrupt ignored
```

In general, these may be ignored.

3.2 Network error messages

The following messages may appear at any time, indicating a network problem. (This list does not cover every possibility.) If the error messages shown appear and the fixes suggested do not solve the problem, you should proceed to the diagnostic system that follows.

`Connection refused`
: The remote server is not running. Check to see if the remote host is down, if it is up in single-user mode, or if its daemons are down.

`Connection timed out`
: This message usually appears after a a long delay (about 2 minutes) following the `remsh` or `rlogin` command. Either the remote host is not up (or is extremely busy) or the appropriate daemon is not up. Check to see if the remote host is down or up in single-user mode. This message can also appear if the system in question is small and has very few network buffers. In this case, the users should try their network commands again later.

`Login incorrect`
: Either the user is trying to use the wrong login name or password, or the `/etc/hosts.equiv` or `.rhosts` file is not set up properly. See `hosts.equiv`(4) in *A/UX Programmer's Reference*. There could also be a corrupt Yellow Pages database; see "Cannot Log In" in the next section.

`m_expand returning 0`
A request to allocate memory to use for `mbufs` was denied. Wait a while and see if the condition clears up before rebooting.

`Permission denied`
The remote system has detected a permissions violation. You can modify either `/etc/hosts.equiv` or `$HOME/.rhosts` on the remote system.

`rcp:`*sys.name*`: No such file or directory`
The `rcp` program cannot find the file or directory to be copied. This can occur when the remote file system has different paths to users' login directories. This is explained in Chapter 3, "Using B-NET," in *A/UX Communications User's Guide*.

rhost`: Host name for your address unknown`
The remote machine does not know the local system's name and address. Modify `/etc/hosts` on the master Yellow Pages server and remake the Yellow Pages database.

rhost:`Unknown host`
The local system does not know the remote machine's name and address. Modify the `/etc/hosts` file on the master Yellow Pages server and remake the Yellow Pages database.

`This machine doesn't exist`
This error message comes only from `talk`.

3.3 Symptoms and tools for analysis

This section presents some of the most common problems you will encounter and describes tools that pinpoint where the problems may lie.

Note that troubleshooting can be considered something of an art, and more experienced administrators will choose or invent different ways to solve problems. This section gives simple, proven suggestions to help solve known problems, but it is by no means complete. Feel free to invent solutions when you are confident, but be careful not to destroy some things while you're trying to fix others.

When you cannot get something done that involves network services, the problem probably lies in one of the following four areas. They are listed with the most likely problem first.

- The network access control policies may not allow the operation (a user or process is trying to access a file system without permission).
- NFS architectural constraints may prevent the operation or cause it to behave unexpectedly. See Chapter 7, "Network Management."
- The client software or hardware may not be working correctly.
- The server software or hardware may not be working correctly.

The following subsections present specific symptoms and probable fixes for these problems in the NFS and Yellow Pages environments.

3.3.1 Cannot reach a machine

If you cannot reach a machine using `rlogin`, `rcp`, or `telnet`, check to see if it's up and running. For example, from `hostname2` type the command

```
/usr/etc/ping hostname1
```

If `hostname1` is running and reachable, the system returns something similar to

```
64 bytes from 128.8.1.1: icmp_seq=0. time=16. ms
64 bytes from 128.8.1.1: icmp_seq=1. time=16. ms
64 bytes from 128.8.1.1: icmp_seq=2. time=16. ms
64 bytes from 128.8.1.1: icmp_seq=3. time=16. ms
64 bytes from 128.8.1.1: icmp_seq=4. time=16. ms
64 bytes from 128.8.1.1: icmp_seq=5. time=16. ms

(interrupt)

----hostname1 PING Statistics----
6 packets transmitted, 6 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 16/16/16
```

If it doesn't return this, the machine could be down, or there could be a break in network service between the two machines. See `ping`(1M).

You may also check to see if the machine is getting Yellow Pages service. Type the command

```
ypmatch hostname2 hosts.byname
```

See `ypmatch`(1). There are three possible responses from this command:

- It could tell you that your machine is not running a `ypbind`. In that case, see "Debugging the Yellow Pages."
- It could tell you that there is no such entry in the database you chose. For example, in the case above, the `hosts.byname` database may contain no entry for a particular host. The lack of such an entry would likely cause the inability to connect over the network to the machine in question. In the case of a missing entry in a Yellow Pages database, you must fix the proper ASCII file on the Yellow Pages master server and then remake the databases from that ASCII file. See "Yellow Pages Administration" in Chapter 7.
- If the entry you request is there, you must look elsewhere for the problem.

3.3.2 Cannot log in

If you know a user belongs on the system and should have an account but cannot log in, check the Yellow Pages password file database on the master server. Sometimes when Yellow Pages databases are made automatically on the Yellow Pages master server, bad databases are generated without an error message. You can use some of the tools mentioned in the preceding section to check Yellow Pages databases. If you have a corrupt database, attempt to remake it on the master server. But before you do, make sure there is enough disk space available on the file system where the Yellow Pages databases are stored. On the target file system, you need twice as much available space as your largest Yellow Pages database. If remaking the Yellow Pages fills up all the space, the program will quit without an error message and leave partial databases where there should be whole ones. This could be why a known user disappears from the system: he or she has been left out of a truncated password file.

Other Yellow Pages databases may be corrupted in the following way. The password file is the first place you're likely to notice problems, but if a machine is dropped from the host database in the manner described

above, you will not be able to communicate with that machine.

Other problems with the services and netgroup databases could also develop. In general, when login, network communication, or network services problems develop suddenly, check the following in this order:

1. Make sure `ypserv` is running on the server and `ypbind` is running on all machines. See Chapter 4, "Adding Yellow Pages Service," and see "Debugging the Yellow Pages" in this chapter.
2. On the master server, check the integrity of all Yellow Pages databases you suspect.
3. Try to remake bad databases. Check first to make sure there's ample disk space.

A good tool to use for checking Yellow Pages databases is `yppoll`. Type

```
/usr/etc/yp/yppoll -h hostname1 hosts.byname
```

where `hostname1` is a Yellow Pages server and `hosts.byname` is a Yellow Pages map name. This will return information about the domain name, the order number of the database, and what the master server is for the database. If you run this before and after you remake a database, you should see an increase in the order number. If this number does not increase, there is a problem. Remember to run `yppoll` on each Yellow Pages server, both slaves and master.

3.3.3 A "`Connection timed out`" message

You see the message

```
Connection timed out
```

when you try to access an NFS soft-mounted file whose server is dead. See Chapter 3, "Initializing NFS," for an explanation of hard and soft NFS mounts; a hard-mounted file system will act differently. You can use the command

```
/usr/etc/rpcinfo -p hostname1
```

(where `hostname1` is a Yellow Pages server) to get information about the state of the NFS server. You may have to fix the server by

restarting processes there. See `rpcinfo`(1M) and "Debugging the NFS" in this chapter for a detailed explanation of the steps.

3.3.4 Cannot NFS-mount a file system

This assumes you are not expressly prevented from mounting the file system for security or other reasons; in other words, you have the appropriate privileges to mount the system. Type the command

```
/usr/etc/showmount -e hostname1
```

where `hostname1` is an NFS server. The output from the `showmount` command shows what file systems are mountable, and by what machines. If your machine is not shown where you expect, it must be added to the `/etc/exports` file on the NFS server machine. (Only the system administrator (`root`) on the server can make this change.) Or if there is a netgroup database that the server machine uses in its `/etc/exports` file, and your machine should be in it, add your machine name to it and remake the database.

3.3.5 Things work, but slowly

This problem could be due to either a heavy load on the network or CPU-intensive jobs running on your server or client machine. There are several tools for monitoring both network traffic and machine load; the most useful of these for network traffic is `netstat`. See "Debugging Network Problems" in Chapter 8 and `netstat`(1N).

4. Debugging the network services

This section is divided into five parts. The first three pertain to debugging a specific part of the network: the network hardware, the NFS service, and the Yellow Pages service. The last two parts deal with specific problems that can develop and error messages that can be generated in the network environment.

4.1 Debugging the network hardware

From time to time, most networks will have problems. Always check your network connections first. On networks such as the Ethernet, a loose cable tap or misplaced transceiver cable can result in severely deteriorated service. The `netstat` program may assist in tracking down hardware malfunctions. In particular, look at the `-i` and `-s` options on the manual page.

If you believe you have a faulty coaxial Ethernet cable, test it to make sure it measures about 50 ohms. See "Hardware Checks" in this section.

If you suspect a `routed` daemon malfunction, you may log its actions—and even all the packet transfers. To create a log file of routing daemon actions, just supply a filename when you start up the daemon; for example,

```
/etc/in.routed /etc/routerlog
```

Whenever a route is added, deleted, or modified, a log of the action and a history of the previous packets set and received will be printed in the log file. To force full packet tracking, specify the `-t` option on the above command line.

> *Note:* Beware! On a busy network, the `-t` option will generate almost constant output. See `routed`(1M) in *A/UX System Administrator's Reference* for more detailed information on daemon options.

Sometimes, even after carefully hooking up your machines, you will have a problem starting up your network and will get the message

```
Connection timed out
```

when trying to use `rlogin` between machines.

The following subsections describe some suggested corrective actions. After each step, test the net again.

4.1.1 Software checks

1. Check `/etc/hosts` on the Yellow Pages master server machine (or the local host if Yellow Pages is not used on your net) to make sure that the entries are correct and up-to-date. Make sure a correct copy has been sent to all Yellow Pages slave servers.
2. Check the accuracy of the information in the `/etc/HOSTNAME` and `/etc/NETADDRS` files.

3. Try `rlogin` to your local host or perform the loopback test. Make sure the network daemon `inetd` is running on the machines that want to talk to each other.

```
ps -e | grep inetd
```

4. `netstat` with the `-i` option tells you how many packets a machine thinks it is transmitting and receiving on each local network. For example, on a server you may see the input packet count increasing each time a client tries to boot, whereas the output packet count remains steady. This suggests that the server is seeing the request packets from the client, but does not respond to them. This might be caused by an incorrect address in `/etc/hosts`. If the input packet count is steady, the machine does not see the packets at all, which suggests a different type of failure, possibly a hardware problem.

4.1.2 Hardware checks

1. With its host system powered on, after a while each transceiver should feel slightly warm to the touch. A cold transceiver is probably not receiving power. This could indicate one of several things: a loose connection on either end of the transceiver cable, a loose connection of the internal Ethernet cable to the Ethernet board, or a faulty cable, transceiver (less likely), or board (even less likely).
2. Check all Ethernet coaxial and transceiver cable connections.
3. Is the network terminated on both ends? It should be. Unscrew one of the terminators and use an ohm meter to test resistance across the coaxial connector where you just unscrewed the terminator (use the pin "inside" the N-connector for signal, and the housing for ground). You should measure about 50 ohms. If you get something other than 50 ohms, your cable may be damaged. This check is particularly pertinent if you use a clamp-on ("vampire clamp") type of transceiver; they tend to short-circuit the Ethernet coaxial cable.
4. Remove one of the terminators and try operating the network. You should get error messages on every machine, something like

```
Ethernet transmission error
```

If a machine just continues to give its previous error (`Connection timed out`), it may not be connected correctly to its transceiver. The problem could be loose connections or a faulty connector, cable, transceiver, or Ethernet board.

5. Try swapping transceivers and transceiver cables. If spare Ethernet boards are available, try swapping boards. Even if there are only two machines on the net, exchanging parts may be informative.

4.2 Debugging the NFS

The process of locating trouble in an NFS installation is complicated by the fact that the server is not necessarily a UNIX® machine.

4.2.1 Remote mount failure: Hard mount vs. soft mount

When a remote mount fails because of a problem on the NFS server machine or on the network itself, local programs that access hard-mounted remote files will fail differently from those that access soft-mounted remote files.

Programs accessing hard-mounted remote file systems will keep trying until the server responds again. From the user's point of view, they will simply hang, as if the server machine were very slow. When the server comes back up again, their program will pick up where it left off. If the server crashes while you are attempting to remotely mount its file systems, a hard mount will hang (just like any other program) and you should see a message

`mount:` *host-name*`:`*/directory* `server not responding: RPC_PMAP_FAILURE – RPC_TIMED_OUT`

and some other information, depending on the condition of the server. If you have specified the `bg` option in `/etc/fstab`, `mount` will also print

`mount: backgrounding` */directory*

You will not be able to interrupt the hanging program, although you can reboot and delete the remote mounts from `/etc/fstab` while in single-user mode.

Programs accessing soft-mounted remote file systems return an error when the server is unreachable. From a user's point of view, an A/UX program accessing a remotely mounted file will usually abort, and if that program checks return conditions on file system operations, the user will see the message

```
Connection timed out
```

on the terminal screen. Unfortunately, many A/UX programs do not check such return conditions, in which case a message will be displayed only on the system console.

4.2.2 Checking the NFS server

Whether you have used a hard mount or a soft one, if you think the server has crashed, use the command (on the client machine)

```
/usr/etc/rpcinfo -p host-name
```

(where *host-name* is the server machine) to check if the server is up. If the server is up, this command will list program, version, protocol, and port numbers:

```
program   vers   proto   port

 10004      2     udp     1027   ypserv
 10004      2     tcp     1024   ypserv
 10004      1     udp     1027   ypserv
 10004      1     tcp     1024   ypserv
 10007      2     tcp     1025   ypbind
 10007      2     udp     1035   ypbind
 10007      1     tcp     1025   ypbind
 10007      1     udp     1035   ypbind
 10003      2     udp     2049   nfs
 10012      1     udp     1111   sprayd
 10005      1     udp     1115   mountd
 10008      1     udp     1117   walld
 10002      1     udp     1119   rusersd
 10002      2     udp     1119   rusersd
 10001      1     udp     1122   rstatd
 10001      2     udp     1122   rstatd
 10001      3     udp     1122   rstatd
```

You can also check whether the remote `mountd` daemon is running with the command (on the client machine)

```
/usr/etc/rpcinfo -u host-name mountd 1
```

where *host-name* is the server machine. If the server is up and its `mountd` daemon is running, this will print

```
program 100005 version 1 ready and waiting
```

If the `rpcinfo` commands fail, log in to the server's console and see if it is OK. If the server is alive but your machine cannot reach it, you should check the Ethernet connections between your machine and the server.

Note: The hardware devices may be physically (but not electrically) connected. In this case, disconnect and then reconnect them.

4.2.3 Checking the NFS client daemons

If the server and network are OK, return to the client machine and use the command

```
ps -ef
```

to check your client daemons. At least a `portmap` daemon and several `biod` daemons should be running.

If the client daemons are running and the server and network connections are OK, check the appropriate subsection.

4.3 Debugging the Yellow Pages

This section describes how to troubleshoot and resolve problems associated with the Yellow Pages services.

4.3.1 Checking `ypserv` and `ypbind`

User processes may also hang when the local `ypbind` process is unable to communicate with `ypserv` in the current domain. If commands hang but you can still start new commands, check the system console for a message such as

```
yp: server not responding for domain name.
Still trying
```

where *name* is the domain name. If this problem is common to all or most of the machines on the network, check the Yellow Pages server machines. If everything is OK on the server machines, `ypserv` may not be able to respond to `ypbind` requests within the timeout period. This can be caused by a heavily loaded network or heavily loaded Yellow Pages server machines, in which case the problem will disappear as soon as the load decreases.

If the local system is the only one experiencing the problem, check that there is at least one Yellow Pages server for your machine's domain running on your local net. Note that two or more Yellow Pages servers in a domain will improve availability and response characteristics for Yellow Pages services.

4.3.2 Checking the Yellow Pages server

When you suspect a problem with a Yellow Pages server, first check that all servers are up. If one or more servers are down, reboot the

machines. Use the command (on each server machine)

```
ps -ef | grep yp
```

to look for the `ypserv` and `ypbind` processes. If the server's `ypserv` daemon is not running, log in on the server as `root` and restart it

```
/etc/ypserv
```

If the `ypbind` process is not running, restart it

```
/etc/ypbind
```

If the server is up and both `ypserv` and `ypbind` are running, check (on the server) whether `ypserv` is hung, by using the command

```
ypwhich
```

If `ypwhich` returns no answer, the `ypserv` daemon is probably hung. On the server machine, kill it and restart it:

```
kill -9 PID
/etc/ypserv
```

If everything is OK on the server machines and both `ypserv` and `ypbind` are running normally, `ypserv` may not be able to respond to `ypbind` requests within the timeout period. This can be caused by a heavily loaded network or heavily loaded Yellow Pages server machines, in which case the problem will disappear as soon as the load decreases. If this occurs often, you should add more Yellow Pages servers to the network to improve availability and response characteristics for Yellow Pages services.

4.3.3 Checking the Yellow Pages client

First check that `ypbind` is running (as shown in "Checking the Yellow Pages Server"). If it is not running, restart it. If `ypbind` is running, check that the domain name agrees on the client and at least one server machine on the local network. From the client machine, use the commands

```
domainname
remsh hostname1 domainname
```

where `hostname1` is a Yellow Pages server. If the Yellow Pages client machine is not behaving properly, this command will probably fail or hang. If they do not agree, edit `/etc/HOSTNAME` and reboot the appropriate system to set the name correctly and then reboot the client system.

If `domainname` agrees on the client and server, check that the remote host you specified is in the local network, not on another accessible network. There must be at least one Yellow Pages server for your machine's domain running on your local net. Using two or more servers improves response time.

4.4 Specific problems in the network environment

This section describes how to resolve specific problems that might occur on your network.

4.4.1 Remote mount hangs during boot

If your machine comes up part way after a boot but hangs where it would normally be doing remote mounts, probably either one or more servers are down or your network connection is bad.

If the server is down, reboot it. If the server machine cannot be rebooted for some reason, reboot the client machine in single-user mode, mount the file system you need to use a text editor, make a copy of `/etc/fstab`, and edit `/etc/fstab` to remove the remote mounts to that server. When you bring the system up in multi-user mode, the problem should disappear.

4.4.2 Processes hang

If programs hang while doing file-related work, the NFS server machine may be down. If your file systems are hard-mounted, you may see this message on the system console:

`NFS server` *host-name* `not responding, still trying`

where *host-name* is the name of the server machine. This is probably a problem with one of your NFS servers or with the Ethernet.

If your machine hangs completely, check the servers from which you have mounted. If one or more of them are down, reboot those systems. When the server comes back up, the client programs will continue automatically, and they will not even know that the server was down.

No files should be lost.

If all of the servers are running, check whether other clients using the same servers are having trouble. If the other machines seem to be OK, check the Ethernet connections on the client machine.

If more than one machine is having problems getting service, it is probably a problem with the server. It could be the `nfsd` daemon or the server's network connection. Log in to the server and do a `ps` to see if `nfsd` is running and accumulating CPU time. If it is not, you may be able to kill and then restart `nfsd`:

```
kill -9 PID1 PID2 PID3 PID4
/etc/nfsd 4
```

If this does not work, reboot the server.

4.4.3 Processes time out

If programs time out while doing file-related work, the NFS server machine may be down. If your file systems are soft-mounted, you may see an error message on users' terminals. If a soft-mounted server goes down, programs accessing it may report errors, but other work on the system should not be affected.

If all of the servers are running, check whether other clients using the same servers are having trouble. If the other machines seem to be OK, check the Ethernet connections on the client and server machines.

If more than one machine is having problems getting service, it is probably a problem with the server's `nfsd` daemon. Log in to the server and enter a `ps` command to see if `nfsd` is running and accumulating CPU time. If it is not, you may be able to kill and then restart `nfsd`:

```
kill -9 PID1 PID2 PID3 PID4
/etc/nfsd 4
```

If this does not work, reboot the server.

4.4.4 Things work, but slowly

If access to remote files seems unusually slow, log in to the server and use the `ps` command to check for a daemon that is respawning improperly, a bad TTY line, and so on.

If the server seems OK and other machines are getting a good response, issue the following command on the client machine to make sure the client's block I/O daemons (that is, `biod`) are running:

```
ps -ef | grep biod
```

If the `biod` daemons on the client machine are not running, you can try to restart them:

```
/etc/biod 4
```

To determine whether the `biod` daemons are hung, do a `ps` as above, then copy a large remote file and do another `ps` command. If the `biod`s don't accumulate CPU time or if the copy hangs, they are probably hung. Kill the processes and then restart the `biod` daemons. If `biod` is OK, check your Ethernet connection.

On the client machine, the command

```
netstat -i
```

will tell you if you are dropping packets. The commands

```
/usr/etc/nfsstat -c
/usr/etc/nfsstat -s
```

can tell if the client or server, respectively, is doing too much retransmitting. (A retransmission rate of 5 percent is considered high.) Excessive retransmission usually indicates a bad Ethernet board, a bad Ethernet tap, a mismatch between board and tap, or a mismatch between Ethernet boards on the server and client machines.

4.4.5 Yellow Pages errors

Propagation failures can be caused by errors in the databases used by the Yellow Pages themselves. Make sure that all Yellow Pages servers are mentioned in the `ypservers` map. Also make sure that all Yellow Pages servers have entries in the `hst.nm` map in both domains.

4.4.6 Commands hang on Yellow Pages client

When commands hang, but the system seems to be OK and you can start new commands, you may see a console message such as

```
yp: server not responding for domain name.
Still trying
```

where *name* is the domain name. This message indicates that the local `ypbind` process is unable to communicate with `ypserv` in the specified domain.

Check that the local `ypbind` process is running (if not, restart it), and check that the local domain name is the same as the domain name on at least one Yellow Pages server on the local network.

4.4.7 Cannot log in

If the Yellow Pages are unavailable on a system, users' passwords may be inaccessible. This usually means that `ypbind` is not running. Check that the local `ypbind` process is running. If it is not, restart it.

If `ypbind` is running, check that at least one `ypserv` process is running in the current domain on the local network.

4.4.8 Yellow Pages commands terminate with a message

Some Yellow Pages commands print more specific error messages. For example, trying the following commands on the client machine produces the subsequent error messages:

```
$ ypcat passwd
```

```
ypcat:can't bind to Yellow Pages server for domain name.
Reason: can't communicate with ypbind
```

If any of these symptoms occurs, try a `ps` command on the client machine to check for `ypbind`; for example,

```
ps -ef | grep ypbind
```

If you do not find `ypbind`, restart it:

```
/etc/ypbind
```

When you have restarted `ypbind`, the Yellow Pages problems should disappear.

4.4.9 Directory listing reports numbers

If you do an `ls -l` on a directory that contains files owned by users who are not in the local machine's `/etc/passwd` file, it may return

a listing such as

```
$ ls -l dir

total 191
-rw-rw-rw-    joe   prog    44    Mar  4   6:08 test1
-rw-rw-rw-    125     12   997    Mar  9   3:00 test2
```

If the `ls -l` reports owners who are not in the local machine's `/etc/passwd` file as numbers rather than names, it indicates that the Yellow Pages service is not working.

Check that the local `ypbind` process is running, and that at least one `ypserv` process is accessible in the current domain.

4.4.10 `ypbind` crashes

If `ypbind` crashes almost immediately each time it is started, you should look for a problem in some other part of the system. Try the command

```
/usr/etc/rpcinfo -p
```

on the client machine to see if the `portmap` daemon is running:

```
program   vers   proto   port

 10004      2     udp     1027   ypserv
 10004      2     tcp     1024   ypserv
 10004      1     udp     1027   ypserv
 10004      1     tcp     1024   ypserv
 10007      2     tcp     1025   ypbind
 10007      2     udp     1035   ypbind
 10007      1     tcp     1025   ypbind
 10007      1     udp     1035   ypbind
 10003      2     udp     2049   nfs
 10012      1     udp     1111   sprayd
 10005      1     udp     1115   mountd
 10008      1     udp     1117   walld
 10002      1     udp     1119   rusersd
 10002      2     udp     1119   rusersd
 10001      1     udp     1122   rstatd
 10001      2     udp     1122   rstatd
 10001      3     udp     1122   rstatd
```

The port numbers will be different for each system. In this case, the number representing the `rpc.mountd` process is `100005`. (`100007` represents the `ypbind` process, and `100004` represents the `ypserv` process.)

If the `portmap` daemon is not running, `rpcinfo` will fail with the message

```
rpcinfo: can't contact portmapper: RPC_SYSTEM_ERROR -
        Connection refused
```

In this case, you should restart the portmapper and then kill any RPC daemons that are running (for example, `rpc.mountd` and `rpc.rstatd`):

```
/etc/portmap
kill -9 PID1 PID2 PID3 PID4
```

Note: This method relies on `init` respawning the daemon

after it is killed, which it will do if `/etc/inittab` has been modified as shown in Chapter 3, "Initializing NFS."

If `portmap` will not stay up or behaves strangely, look for more fundamental problems. Check the network software. You may be able to talk to the `portmap` daemon on your machine by using the `/usr/etc/rpcinfo` command from another machine that is operating normally. If `ypbind` doesn't show up, reboot the machine.

If `ypbind` is there and changes each time you try to restart `/etc/ypbind`, reboot the system, even if the `portmap` daemon is up.

4.4.11 `ypserv` crashes

If the `ypserv` process crashes almost immediately and will not stay up even with repeated activations, the debugging process is the same as that described in "`ypbind` Crashes."

4.5 Error messages in the network environment

`/etc/mtab: No such file or directory`
: The mounted file system table is kept in the file `/etc/mtab`. This file must exist before `mount` can succeed. Edit the file `/etc/fstab` to update `/etc/mtab`.

`mount:` *host*`:` *file-system* `already mounted`
: The file system that you are trying to mount is already mounted or there is an incorrect entry for it in `/etc/fstab`.

`mount:` *host*`:` *file-system* `Block device required`
: You probably left the `rhost` part off of the remote `mount` request. The `mount` command assumes you are doing a local mount, unless it sees a colon in the file system name or the file system type is `nfs` in `/etc/fstab`.

`mount:` *host*`:` *file-system* `not found in /etc/fstab`
: If `mount` is called with only a directory or file system name (but not both), it looks in `/etc/fstab` for an entry whose file system or directory field matched the argument. For example, the command

```
mount /rickusr
```

searches `/etc/fstab` for a line that has a directory name field of `/rickusr`. If it finds an entry, such as

```
rick:/usr /rickusr nfs rw,hard 0 0
```

it will do the mount as if you had typed the full command. If you see this message, it means the argument you gave `mount` was not in any of the entries in `/etc/fstab`.

`/etc/fstab: No such file or directory`
The `mount` command tried to look up the name in `/etc/fstab`, but there was no such file.

host`:` *file-system* `not in hosts database`
This means either that the Yellow Pages could not find the host name you gave in the remote `mount` command or that the Yellow Pages daemon (`ypbind`) is down on your machine. First check the spelling and the placement of the colon in your `mount` call. If it looks OK, make sure that `ypbind` is running, by typing

```
ps -ef | grep ypbind
```

Try `remsh` or `rcp` to some other machine. If this also fails, your `ypbind` is probably down or hung. If you get this message for only one host name, it means that the `/etc/hosts` entry on the Yellow Pages server needs to be checked. See "Specific Problems in the Network Environment."

`mount:Directory path must begin with /`
The second argument to `mount` is the path of the directory to be covered. This must be an absolute path.

`mount:`*host*`: file-system server not responding: RPC_PMAP_FAILURE - RPC_TIMED_OUT`
Either the server you are trying to mount from is down, or its portmapper is dead or hung. Try logging in to that machine. If you can log in, try running

```
rpcinfo -p host-name
```

You should get a list of registered program numbers. If you do not get such a list, restart the portmapper. Note that restarting the portmapper requires that you then kill and restart `ypbind` as well. After you have killed the `portmap`, `ypbind`, and `inetd` daemons (by using `kill -9` *PID*), restart them with

```
/etc/portmap
/etc/ypbind
/etc/inetd
```

If you do not want to do this, you can just reboot the server.

If you cannot use `rlogin` to the server but the server is up, you should check your Ethernet connection by trying `rlogin` to some other machine. You should also check the server's Ethernet connection.

`mount:` *host*`:` *file-system* `server not responding: RPC_PROG_NOT_REGISTERED`

This means that `mount` got through to the portmapper, but the NFS mount daemon (`rpc.mountd`) was not registered. Go to the server and be sure that `/usr/etc/rpc.mountd` exists and is executable. Look in `/etc/servers` to make sure there is an entry for `rpc.mountd`. Look in `/etc/inittab` to make sure `/etc/inetd` is enabled, and check that it's running.

`mount:` *host*`:` *file-system*`: No such file or directory`

Either the remote directory or the local directory does not exist. Check spelling and try to run `ls` on both directories.

`mount: access denied for` *host*`:` *file-system*

Your machine name is not in the export list for the file system that you want to mount from the server. You can get a list of the server's exported file systems by running

```
showmount -e host-name
```

If the file system you want is not in the list, or if your machine name or netgroup name is not in the user list for the file system,

log in to the server and check the `/etc/exports` file for the correct file system entry. A file system name that appears in the `/etc/exports` file but not in the output from `showmount` indicates a failure in `mountd`. Either `mountd` could not parse that line in the file or could not find the file system, or the file system name was not a locally mounted file system. See `exports`(4). If `exports` seems OK, check the server's `ypbind` daemon, as it may be down or hung.

`mount:` *host*`:` *file-system* `Permission denied`
This message is a generic indication that an attempt to authenticate failed on the server; hence it could imply that one of several conditions holds. It could be simply that you are not in the export list (see the preceding message) or that the server could not figure out who you are (`ypbind` is dead), or it could be that the server does not believe that you are who you say you are. For the first two cases, check the server's `/etc/exports` and `ypbind`, and fix them if necessary. In the last case, change your host name (by using the `hostname` command). Then, in all cases, retry the mount.

`mount:` *host*`:` *file-system* `Not a directory`
Either the remote path or the local path is not a directory. Check spelling and try to run `ls` on both directories.

`mount:` *host*`:` *file-system* `Not owner`
You have to do the mount as `root` on your machine, because it affects not just you but the file system for the whole machine.

Chapter 10
Installing and Administering AppleTalk

Contents

1. Overview of AppleTalk under A/UX 1

2. The printer interface 2
   - 2.1 Setting up the printer spooler 2
   - 2.2 Printing files with `lp` 5
   - 2.3 Choosing a printer for each user 6
   - 2.4 Printing files with `at_printer` 7

3. Summary of AppleTalk commands 8

Chapter 10

Installing and Administering AppleTalk

1. Overview of AppleTalk under A/UX

This chapter describes the A/UX implementation of AppleTalk and is designed for network administrators who want to set up an AppleTalk network.

AppleTalk can work concurrently with the B-NET, NFS, and Yellow Pages software described in the previous chapters of this guide. AppleTalk gives you the capability of putting your A/UX system on a network of computers connected by low-cost LocalTalk cables for the purpose of sharing an Apple LaserWriter printer.

Because of the high processing requirements of both AppleTalk and the A/UX multitasking operating system, A/UX requires a co-processing board for AppleTalk to work effectively. Contact your Apple representative for a list of boards that are available from various vendors.

Instructions for setting up an AppleTalk network with LocalTalk cables and connectors can be found in *LocalTalk Cable System Owner's Guide* from Apple Computer, Inc., part number 030-0043-A. For instructions on installing AppleTalk under A/UX, refer to the owner's guide that accompanies your AppleTalk board. Very briefly, installation generally consists of the following steps:

1. Shut down A/UX.
2. Remove the cover from your computer.
3. Insert the AppleTalk board into an available NuBus slot.
4. Replace the cover.
5. Attach a LocalTalk cable to the connector on the back of the AppleTalk board.

6. Restart your computer and bring A/UX up to single-user mode.
7. Install the software from any accompnaying floppy disks.
8. Run the `newunix` and `autoconfig` commands to build a new kernal that integrates the device driver for the AppleTalk board.
9. Reboot A/UX and bring it back to multiuser mode.

Specific, detailed installation instructions are include in the owner's guide for your AppleTalk product. The rest of this chapter describes how to configure an AppleTalk printer interface under A/UX.

2. The printer interface

Your computer is set up so that an ImageWriter II (`iw2`) is the default printer. It functions as a serial device. To change the default printer to an AppleTalk printer, follow the steps in the sections below.

> *Note:* To use an ImageWriter II with AppleTalk, you need a special conversion kit (see your Apple representative for details). Without this kit, an ImageWriter II can function only as a serial printer.

2.1 Setting up the printer spooler

To use the A/UX `lp` spooler to print jobs across the AppleTalk network, you need to provide certain information to `lp`. Follow these steps:

1. Log in as `root` or `lp`.
2. Move to the print spool directory by entering

   ```
   cd /usr/spool/lp
   ```

3. Check whether `lpsched` is running by entering

   ```
   lpstat -t
   ```

   If it is running, you will see

   ```
   scheduler is running
   ```

   at the beginning of the screen message that appears. If you do,

turn off the scheduler by entering

```
/usr/lib/lpshut
```

When you turn off the scheduler, it should remove two files from the `/usr/spool/lp` directory. List the directory to see whether the files `FIFO` and `SCHEDLOCK` are there. If they are, remove them.

```
rm FIFO SCHEDLOCK
```

To check that all is well at this point, retype

```
lpstat -t
```

The response should be

```
scheduler is not running
no system default destination
```

4. Now check the `model` subdirectory for a file named `at_interface`.

   ```
   ls -l model/at_interface
   ```

   The response should be similar to this:

   ```
   -rw-r--r--  1  lp   lp  6117 Aug 1  18:44 at_interface
   ```

   This is a shell script interface for LaserWriters (or ImageWriters) on the AppleTalk network. You can modify this script if your network requires it.

5. You now need to supply three pieces of information to the `lpadmin` command:

   - For the `-p` flag option, specify the name of the printer (`bill`, in the example).
   - For the `-m` flag option, specify the name of the interface program in the `model` directory (`at_interface`, in the example).
   - For the `-v` flag option, specify the name of the special device for output. The `lp` spooler usually requires a TTY device, such as `/dev/tty01`. However, to use

AppleTalk you should supply either `/dev/null` or `/dev/console`. (This is because `at_interface` execs another program for its output rather than sending it to standard output as most `lp` interface scripts do.) If you want to see error output, you should supply `/dev/console`.

Type

```
/usr/lib/lpadmin -pbill -mat_interface -v/dev/console
```

The `-m` flag option copies the specified interface program from the `model` subdirectory into the `interface` subdirectory and renames it to the printer's name.

6. To check that all is well at this point, list the `interface` subdirectory

```
ls -l interface
```

The response should be similiar to this:

```
-rw-r--r--  1  lp   lp  6117 Nov 1  09:25 iw2
-rw-r--r--  1  lp   lp  6117 Nov 1  09:25 bill
```

Then type

```
lpstat -t
```

The response should be

```
scheduler is not running
system default destination: iw2
device for iw2: /dev/iw2
device for bill: /dev/console
iw2 accepting requests since date
bill not accepting requests
   new destination
printer iw2 s idle.  enabled since date
printer bill disabled since date
   new printer
```

7. To use `bill` as the default destination for `lp`, type

```
/usr/lib/lpadmin -dbill
```

8. Restart the scheduler by typing

   ```
   /usr/lib/lpsched
   ```

   The response to `lpstat -t` will now include

   ```
   scheduler is running
   system default destination: bill
   ```

9. Tell the spooler to accept print requests by typing

   ```
   /usr/lib/accept bill
   ```

   The response to `lpstat -t` will now include

   ```
   destination "bill" now accepting requests
   ```

10. Tell the printer to print the requests queued in the spooler by typing

    ```
    enable bill
    ```

    The response to `lpstat -t` should now read:

    ```
    scheduler is running
    system default destination: bill
    device for iw2: /dev/iw2
    device for bill: /dev/console
    iw2 accepting requests since date
    bill accepting requests since date
    printer iw2 is idle.  enabled since date
    printer bill is idle.  enabled since date
    ```

2.2 Printing files with `lp`

To print ASCII or PostScript files on printers attached to the `lp` spooling system, use the command

`lp -d`*printer file*

or

`cat` *file* `| lp -d`*printer*

where *printer* is the printer's `lp` name.

To print formatted `troff` output files, use the command

```
cat file | psdit | lp
```

To print formatted `psroff` output files, use the command

```
cat file | lp
```

Note: When you use the `psroff` command to format text, the file is automatically run through `psdit`.

You can print an ASCII file in two ways:

```
cat file | lp
```

or

```
enscript -p- -r2 file | lp
```

The `enscript` command produces "landscaped" output; that is, text printed horizontally on the page. See `enscript`(1) in *A/UX Command Reference* for details.

See `/usr/lib/ps` for the Adobe PostScript programs available with A/UX.

2.3 Choosing a printer for each user

Each user may choose his or her own default printer by using the `at_cho_prn` command. Alternatively, the system administrator can set the default printer for a user:

1. Check that the LaserWriter is turned on.
2. Use the `cd` command to move to that user's home directory.
3. Enter

   ```
   at_cho_prn
   ```

   This displays information for every printer in the AppleTalk network local zone; for example,

```
ITEM OBJECT  TYPE          ZONE  NET     NODE  SOCKET
1:   bill    LaserWriter   *     0x1111  0x11   0xca

P: ITEM number (0 to make no selection)?
```

where the headings mean the following:

ITEM	This is an identification number for the entire entry.
OBJECT	This is the printer's name.
TYPE	This is the type of printer. In the current distribution, this will always be `LaserWriter`.
ZONE	This is the zone in which the printer resides. An asterisk (*) indicates the local zone.
NET	If more than one AppleTalk network is connected by a 4-digit hexadecimal number.
NODE	This is the system's node number on the network; a 2-digit hexadecimal number. It is assigned automatically when the network is set up.
SOCKET	This is the socket number for this service.

In this case, because there is only one item displayed, type

```
1
```

(for ITEM 1) and press RETURN. This creates an entry in the file `/usr/lib/PrinterChoices` with the user identification number (UID) and the chosen default printer.

See `at_cho_prn`(1) in *A/UX Command Reference* for more information.

2.4 Printing files with `at_printer`

The `at_printer` command sends data directly to a PAP server, such as a LaserWriter. This script is useful when there is a single user on a system or when the `lp` spooler is not configured.

The `at_printer` script is faster than the spooler because it sends files directly to the output device. However, with the spooler, print requests from an individual user can be queued; in contrast, with

`at_printer`, requests can be issued only one at a time. To print formatted `troff` output files, use the command

```
cat file | psdit | at_printer
```

This sends *file* to the default printer chosen with `at_cho_prn`.

To print plain ASCII text, use the command

```
cat file | pstext | at_printer
```

3. Summary of AppleTalk commands

`at_cho_prn`	Chooses a user's default printer on the LocalTalk network. See `at_cho_prn`(1) in *A/UX Command Reference*.
`at_nvelkup`	Looks up NVEs (network visible entities) registered on the LocalTalk network. See `at_nvelkup`(1) in *A/UX Command Reference*.
`at_printer`	Copies data to a remote PAP (Printer Access Protocol) server. See `at_printer`(1) in *A/UX Command Reference*.
`at_server`	Generic PAP server. See `at_server`(1) in *A/UX Command Reference*.
`at_status`	Returns the status of a PAP server. See `at_server`(1) in *A/UX Command Reference*.
`appletalk`	Configures and views AppleTalk network interfaces. See `appletalk`(1M) in *A/UX System Administrator's Reference*.
`newunix`	Prepares for a new kernel configuration. See `newunix`(1M) in *A/UX System Administrator's Reference*.

Appendix A
Implementing a `sendmail` Facility

Contents

1. Basic installation 2
   1.1 Off-the-shelf configurations 2
   1.2 Installing with the makefile 4
   1.3 Installing by hand 4
      1.3.1 `lib/libsys.a` 4
      1.3.2 `/usr/lib/sendmail` 5
      1.3.3 `/usr/lib/sendmail.cf` 5
      1.3.4 `/usr/ucb/newaliases` 5
      1.3.5 `/usr/spool/mqueue` 5
      1.3.6 `/usr/lib/aliases*` 5
      1.3.7 `/usr/lib/sendmail.fc` 6
      1.3.8 `/etc/rc` 6
      1.3.9 `/usr/lib/sendmail.hf` 7
      1.3.10 `/usr/lib/sendmail.st` 7
      1.3.11 `/usr/ucb/newaliases` 7
      1.3.12 `/usr/ucb/mailq` 7
2. Normal operations 8
   2.1 Quick configuration startup 8
   2.2 The mail queue 8
      2.2.1 Printing the queue 8
      2.2.2 Format of queue files 8
      2.2.3 Forcing the queue 10
   2.3 The alias database 11
      2.3.1 Rebuilding the alias database 12
      2.3.2 Potential problems 12
      2.3.3 List owners 13
   2.4 Per-user forwarding (`.forward` files) 13
   2.5 Special header lines 14

Appendix A

Implementing a `sendmail` Facility

This appendix was originally printed as the *Sendmail Installation and Operation Guide Version 5.8*, written by Eric Allman of the Computer Science Research Group at the University of California, Berkeley, and is reproduced here for your reference with permission. Specific instructions for installing and operating the version of `sendmail` distributed with A/UX are offered in "A Network Mail System," in Chapter 7.

`sendmail` implements a general-purpose internetwork mail routing facility under the UNIX operating system. It is not tied to any one transport protocol; its function may be likened to a crossbar switch, relaying messages from one domain into another. In the process, it can do a limited amount of message header editing to put the message into a format that is appropriate for the receiving domain. All of this is done under the control of a configuration file.

Because of the flexibility requirements for `sendmail`, the configuration file can seem somewhat unapproachable. However, there are only a few basic configurations for most sites, for which standard configuration files have been supplied. Most other configurations can be built by adjusting an existing configuration file incrementally.

Although `sendmail` is intended to run without the need for monitoring, it has a number of features that may be used to monitor or adjust the operation under unusual circumstances. These features are described in this appendix.

Section 1 describes how to do a basic `sendmail` installation. Section 2 explains the day-to-day information you should know to maintain your mail system. If you have a relatively normal site, these two sections should contain sufficient information for you to install `sendmail` and keep it happy. Section 3 describes some parameters that may be safely tweaked. Section 4 has information regarding the

command line arguments. Section 5 contains the nitty-gritty information about the configuration file. This section is for masochists and people who must write their own configuration file. Sections 6 through 10 of this appendix give a brief but detailed explanation of other features.

The references in this appendix are found in the companion paper *Sendmail – An Internetwork Mail Router*, which provides a basic understanding of how the pieces fit together.

1. Basic installation

There are two basic steps to installing `sendmail`. The hard part is to build the configuration table. This is a file that `sendmail` reads when it starts up that describes the mailers it knows about, how to parse addresses, how to rewrite the message header, and what the settings are of various options. Although the configuration table is complex, a configuration can usually be built by adjusting an existing off-the-shelf configuration. The second part of installing `sendmail` is actually doing the installation, that is, creating the necessary files, and so on.

This section describes the installation of `sendmail` assuming you can use one of the existing configurations and that the standard installation parameters are acceptable. All pathnames and examples are given from the root of the `sendmail` subtree, normally `/usr/src/usr.lib/sendmail` on 4.3BSD.

1.1 Off-the-shelf configurations

The configuration files are all in the subdirectories `cf.named` and `cf.hosttable` of the `sendmail` directory. The directory `cf.named` contains configuration files that have been tailored for the name server `named`(1M). These are the configuration files currently used at Berkeley. The configuration files in `cf.hosttable` are some typical ones and the old Berkeley versions from before the name server was used. You should create a symbolic link from `cf` to the directory that you are going to use. For example, to use the name server, enter

```
ln -s cf.named cf
```

The configuration files used at Berkeley are in `m4`(1) format; files with

names ending with `.m4` are `m4` include files, and files with names ending with `.mc` are the master files. Files with names ending with `.cf` are the `m4` processed versions of the corresponding `.mc` file.

Three standard configurations handle the basic cases:

- Arpanet (TCP) sites not running the name server can use `cf.hosttable/arpaproto.cf`. For simple sites, you should be able to use this file without modification. This file is not in `m4` format.
- UUCP (UNIX-to-UNIX System Copy) sites can use `cf.hosttable/uucpproto.cf`. If your UUCP node name is not the same as your system name (as printed by the `hostname` command), you may have to modify the `U` macro. This file is not in `m4` format.
- A group of machines at a single site connected by an Ethernet with only one host connected to the outside world via UUCP is represented by two configuration files: `cf.hosttable/lanroot.mc` should be installed on the host with outside connections, and `cf.hosttable/lanleaf.mc` should be installed on all other hosts. These will require slightly more configuration. First, in both files the `D` macro and `D` class must be adjusted to indicate your local domain. For example, if your company is known as `Muse`, you will want to change both of those accordingly. (As distributed, they are called `XXX`.) Second, in `lanleaf.mc` you will have to change the `R` macro to the name of the root host, that is, the host that runs `lanroot.mc`. For example, they might appear as

  ```
  DDMuse
  CDLOCAL Muse
  DRErato
  ```

  Internally, the root host will be known as `Erato.Muse` and other hosts will be known as `Thalia.Muse`, `Clio.Muse`, and so forth.

The file you need should be copied to a file with the same name as your system; for example,

```
cp uucpproto.cf ucsfcgl.cf
```

This file is now ready for installation as `/usr/lib/sendmail.cf`.

1.2 Installing with the makefile

A makefile exists in the root of the `sendmail` directory that will do all of these steps for a 4.3BSD system. It may have to be slightly tailored for use on other systems.

Before using this makefile, create a symbolic link from `cf` to the directory containing your configuration files. You should also create your configuration file and leave it in the file `cf/`*system*`.cf`, where *system* is the name of your system (that is, what is returned by `hostname`). If you do not have `hostname`, you can use the declaration `HOST=`*system* on the `make` command line. You should also examine the file `md/config.m4` and change the `m4` macros there to reflect any libraries and compilation flags you may need.

The basic installation procedure is to enter

```
make
make install
make installcf
```

in the root directory of the `sendmail` distribution. This will make all binaries and install them in the standard places. The second and third `make` commands must be executed as the superuser (`root`).

1.3 Installing by hand

Along with building a configuration file, you will have to install the `sendmail` startup into your system. If you are doing this installation in conjunction with a regular install, these steps will already be complete. Many of these steps will have to be executed as the superuser (`root`).

1.3.1 `lib/libsys.a`

The library in `lib/libsys.a` contains some routines that should in some sense be part of the system library. These are the system logging routines and the new directory access routines (if required). If you are not running the 4.3BSD directory code and do not have the compatibility routines installed in your system library, execute the command

```
(cd lib; make ndir)
```

This will compile and install the 4.3 compatibility routines in the library. Then enter

```
(cd lib; make)
```

This will recompile and fill the library.

1.3.2 /usr/lib/sendmail

The binary for `sendmail` is located in `/usr/lib`. There is a version available in the source directory that is probably inadequate for your system. Plan to recompile and install the entire system:

```
cd src
make clean
make
cp sendmail /usr/lib
chgrp kmem /usr/lib/sendmail
```

1.3.3 /usr/lib/sendmail.cf

Install the configuration file you created earlier in `/usr/lib/sendmail.cf`:

```
cp cf/system.cf /usr/lib/sendmail.cf
```

1.3.4 /usr/ucb/newaliases

If you are running `delivermail`, it is critical that the `newaliases` command be replaced. This can just be a link to `sendmail`:

```
rm -f /usr/ucb/newaliases
ln /usr/lib/sendmail /usr/ucb/newaliases
```

1.3.5 /usr/spool/mqueue

Create the directory `/usr/spool/mqueue` to hold the mail queue. This directory should be mode 777 unless `sendmail` is run setuid, when `mqueue` should be owned by the `sendmail` owner and mode 755.

1.3.6 /usr/lib/aliases*

The system aliases are held in three files. The file `/usr/lib/aliases` is the master copy. A sample is given in

`lib/aliases` that includes some aliases that *must* be defined:

```
cp lib/aliases /usr/lib/aliases
```

You should extend this file with any aliases that are appropriate to your system.

Normally, `sendmail` looks at a version of these files maintained by the dbm(3X) routines. These are stored in `/usr/lib/aliases.dir` and `/usr/lib/aliases.pag`. You can initially create these as empty files, but they will have to be initialized promptly. These should be mode 666 if you are running a reasonably relaxed system:

```
cp /dev/null /usr/lib/aliases.dir
cp /dev/null /usr/lib/aliases.pag
chmod 666 /usr/lib/aliases.*
newaliases
```

1.3.7 /usr/lib/sendmail.fc

If you intend to install the frozen version of the configuration file (for quick startup), create the file `/usr/lib/sendmail.fc` and initialize it. This step may be safely skipped.

```
cp /dev/null /usr/lib/sendmail.fc
/usr/lib/sendmail -bz
```

1.3.8 /etc/rc

It will be necessary to start up the `sendmail` daemon when your system reboots. This daemon performs two functions: it listens on the SMTP (Simple Mail Transfer Protocol) socket for connections (to receive mail from a remote system), and it processes the queue periodically to ensure that mail gets delivered when hosts come up.

Add the following lines to `/etc/rc` (or `/etc/rc.local` as appropriate) in the area where it is starting up the daemons:

```
if [ -f /usr/lib/sendmail ]; then
   (cd /usr/spool/mqueue; rm -f [lnx]f*)
   /usr/lib/sendmail -bd -q30m &
   echo -n ' sendmail' >/dev/console
fi
```

The `cd` and `rm` commands ensure that all lock files have been removed; extraneous lock files may be left around if the system goes down in the middle of processing a message. The line that actually invokes `sendmail` has two flags: `-bd` causes it to listen on the SMTP port, and `-q30m` causes it to run the queue every half hour.

If you are not running a version of the UNIX system that supports Berkeley TCP/IP, do not include the `-bd` flag option.

1.3.9 /usr/lib/sendmail.hf

This is the help file used by the SMTP `HELP` command. Copy it from `lib/sendmail.hf`:

```
cp lib/sendmail.hf /usr/lib
```

1.3.10 /usr/lib/sendmail.st

If you want to collect statistics about your mail traffic, you should create the file `/usr/lib/sendmail.st`:

```
cp /dev/null /usr/lib/sendmail.st
chmod 666 /usr/lib/sendmail.st
```

This file does not grow. It is printed with the program `aux/mailstats`.

1.3.11 /usr/ucb/newaliases

If `sendmail` is invoked as `newaliases`, it will simulate the `-bi` flag option (that is, it will rebuild the alias database). This should be a link to `/usr/lib/sendmail`.

1.3.12 /usr/ucb/mailq

If `sendmail` is invoked as `mailq`, it will simulate the `-bp` flag option (that is, it will print the contents of the mail queue). This should be a link to `/usr/lib/sendmail`.

2. Normal operations

2.1 Quick configuration startup

A fast version of the configuration file may be set up by using the `-bz` flag option:

```
/usr/lib/sendmail -bz
```

This creates the file `/usr/lib/sendmail.fc` (frozen configuration). This file is an image of `sendmail`'s data space after reading in the configuration file. If this file exists, it is used instead of `/usr/lib/sendmail.cf`. `sendmail.fc` must be rebuilt manually every time `sendmail.cf` is changed.

The frozen configuration file will be ignored if a `-C` flag option is specified or if `sendmail` detects that it is out of date. However, the heuristics are not strong so this should not be trusted.

2.2 The mail queue

The mail queue should be processed transparently. However, you may find that manual intervention is sometimes necessary. For example, if a major host is down for a period of time, the queue may become clogged. Although `sendmail` ought to recover gracefully when the host comes up, you may find performance unacceptable in the meantime.

2.2.1 Printing the queue

The contents of the queue can be printed by using the `mailq` command (or by specifying the `-bp` flag option to `sendmail`):

```
mailq
```

This will produce a listing of the queue IDs, the size of the message, the date the message entered the queue, and the sender and recipients.

2.2.2 Format of queue files

All queue files have the form *xfAA99999*, where *AA99999* is the ID for this file and *x* is a type. The types are

`d` The data file. The message body (excluding the header) is kept in this file.

`l` The lock file. If this file exists, the job is currently being processed, and a queue run will not process the file. For that reason, an extraneous `lf` file can cause a job to apparently disappear (it will not even time out!).

`n` A file that is created when an ID is being created. This is a separate file to ensure that no mail can ever be destroyed because of a race condition. It should exist for no more than a few milliseconds at any given time.

`q` The queue control file. This file contains the information necessary to process the job.

`t` A temporary file. This is an image of the `qf` file when it is being rebuilt. It should be renamed to a `qf` file very quickly.

`x` A transcript file, existing during the life of a session showing everything that happens during that session.

The `qf` file is structured as a series of lines, each beginning with a code letter. The lines are as follows:

`D` The name of the data file. There may be only one of these lines.

`H` A header definition. There may be any number of these lines. The order is important: they represent the order in the final message. These use the same syntax as header definitions in the configuration file.

`R` A recipient address. This will normally be completely aliased, but is actually realiased when the job is processed. There will be one line for each recipient.

`S` The sender address. There may be only one of these lines.

`E` An error address. If any such lines exist, they represent the addresses that should receive error messages.

`T` The job creation time. This is used to compute when to time out the job.

`P` The current message priority. This is used to order the queue. Higher numbers mean lower priorities. The priority changes as the message sits in the queue. The initial priority depends on the

message class and the size of the message.

M A message. This line is printed by the `mailq` command and is generally used to store status information. It can contain any text.

As an example, the following is a queue file sent to `mckusick@calder` and `wnj`:

```
DdfA13557
Seric
T404261372
P132
Rmckusick@calder
Rwnj
H?D?date: 23-Oct-82 15:49:32-PDT (Sat)
H?F?from: eric (Eric Allman)
H?x?full-name: Eric Allman
Hsubject: this is an example message
Hmessage-id: <8209232249.13557@UCBARPA.BERKELEY.ARPA>
Hreceived: by UCBARPA.BERKELEY.ARPA (3.227 [10/22/82])
   id A13557; 23-Oct-82 15:49:32-PDT (Sat)
HTo: mckusick@calder, wnj
```

This shows the name of the data file, the person who sent the message, the submission time (in seconds since January 1, 1970), the message priority, the message class, the recipients, and the headers for the message.

2.2.3 Forcing the queue

`sendmail` should run the queue automatically at intervals. The algorithm is to read and sort the queue, and then attempt to process all jobs in order. When it attempts to run the job, `sendmail` first checks to see if the job is locked. If so, it ignores the job.

There is no attempt to ensure that only one queue processor exists at any time, because there is no guarantee that a job cannot take forever to process. Because of the locking algorithm, it is impossible for one job to freeze the queue. However, an uncooperative recipient host or a program recipient that never returns can accumulate many processes in your system. Unfortunately, there is no way to resolve this without

violating the protocol.

In some cases, a major host going down for a couple of days can create a prohibitively large queue. This will result in `sendmail` spending an inordinate amount of time sorting the queue. This situation can be fixed by moving the queue to a temporary place and creating a new queue. The old queue can be run later when the offending host returns to service.

To do this, move the entire queue directory:

```
cd /usr/spool
mv mqueue omqueue; mkdir mqueue; chmod 777 mqueue
```

You should then kill the existing daemon (because it will still be processing in the old queue directory) and create a new daemon.

To run the old mail queue, run the following command:

```
/usr/lib/sendmail -oQ/usr/spool/omqueue -q
```

The `-oQ` flag specifies an alternate queue directory and the `-q` flag says to just run every job in the queue. You can use the `-v` flag option to watch what is going on.

When the queue is finally emptied, you can remove the directory:

```
rmdir /usr/spool/omqueue
```

2.3 The alias database

The alias database exists in two forms. One is a text form, maintained in the file `/usr/lib/aliases`. The aliases are of the form

name: *name1*, *name2*, ...

Only local names can be aliased; for example,

```
eric@mit-xx: eric@berkeley.EDU
```

will not have the desired effect. Aliases can be continued by starting any continuation lines with a space or a tab. Blank lines and lines beginning with a number sign (#) are comments.

The second form is processed by the `dbm`(3X) library. This form is in the files `/usr/lib/aliases.dir` and

/usr/lib/aliases.pag. This is the form that sendmail actually uses to resolve aliases. This technique improves performance.

2.3.1 Rebuilding the alias database

The dbm version of the database can be rebuilt explicitly by executing the command

```
newaliases
```

This is equivalent to giving sendmail the -bi flag:

```
/usr/lib/sendmail -bi
```

If the D option is specified in the configuration, sendmail will rebuild the alias database automatically if possible when it is out of date. It will do this under one of the following conditions:

- The dbm version of the database is mode 666,

or

- sendmail is running setuid to root.

Auto-rebuild can be dangerous on heavily loaded machines with large alias files; if it might take more than 5 minutes to rebuild the database, there is a chance that several processes will start the rebuild process simultaneously.

2.3.2 Potential problems

A number of problems can occur with the alias database. They all result from a sendmail process accessing the dbm version while it is only partially built. This can happen under two circumstances: one process accesses the database while another process is rebuilding it, or the process rebuilding the database dies (because it is killed or the system crashes) before completing the rebuild.

sendmail has two techniques to try to relieve these problems. First, it ignores interrupts while rebuilding the database; this avoids the problem of someone aborting the process and leaving a partially rebuilt database. Second, at the end of the rebuild it adds an alias of the form

```
@: @
```

(which is not normally legal). Before sendmail will access the database, it checks to ensure that this entry exists.

Note: The `a` option is required in the configuration for this action to occur. This should normally be specified unless you are running `delivermail` in parallel with `sendmail`.

`sendmail` will wait for this entry to appear, at which point it will force a rebuild itself.

Note: The `D` option must be specified in the configuration file for this operation to occur. If the `D` option is not specified, a warning message is generated and `sendmail` continues.

2.3.3 List owners

If an error occurs on sending to a certain address, say *x*, `sendmail` will look for an alias of the form `owner-`*x* to receive the errors. This is typically useful for a mailing list where the submitter of the list has no control over the maintenance of the list itself; in this case, the list maintainer would be the owner of the list. For example,

```
unix-wizards: eric@ucbarpa, wnj@monet,
    nosuchuser, sam@matisse
owner-unix-wizards: eric@ucbarpa
```

would cause `eric@ucbarpa` to get the error that will occur when someone sends to `unix-wizards` because of the inclusion of `nosuchuser` on the list.

2.4 Per-user forwarding (`.forward` files)

As an alternative to the alias database, any user may put a file with the name `.forward` in his or her home directory. If this file exists, `sendmail` redirects mail for that user to the list of addresses in the `.forward` file. For example, if the home directory for user `mckusick` has a `.forward` file with contents

```
mckusick@ernie
kirk@calder
```

any mail arriving for `mckusick` will be redirected to the specified accounts.

2.5 Special header lines

Several header lines have special interpretations defined by the configuration file. Others have interpretations built into `sendmail` that cannot be changed without changing the code. These built-in header lines are described here.

2.5.1 `Return-receipt-to:`

If this header is sent, a message will be sent to any specified addresses when the final delivery is complete, that is, when successfully delivered to a mailer with the `-l` flag (local delivery) set in the mailer descriptor.

2.5.2 `Errors-to:`

If errors occur anywhere during processing, this header will cause error messages to go to the listed addresses rather than to the sender. This is intended for mailing lists.

2.5.3 `Apparently-to:`

If a message comes in with no recipients listed in the message (in a `To:`, `Cc:`, or `Bcc:` line), `sendmail` will add the `Apparently-To:` header line for any recipients it is aware of. (This is not intended as a standard recipient line to warn any recipients that the list is not complete.)

At least one recipient line is required under RFC 822 (see Appendix B for information on obtaining this document).

3. Arguments

The complete list of arguments to `sendmail` is described in detail in Section 6. Some important arguments are described here.

3.1 Queue Interval

The amount of time between forking a process to run through the queue is defined by the `-q` flag. If you run in mode `f` or `a`, this can be relatively large, because it will be relevant only when a host that was down comes back up. If you run in `q` mode, it should be relatively short because it defines the maximum amount of time that a message can sit in the queue.

3.2 Daemon mode

If you allow incoming mail over an IPC connection, you should have a daemon running. This should be set by your `/etc/rc` file by using the `-bd` flag option. The `-bd` flag and the `-q` flag can be combined in one call:

```
/usr/lib/sendmail -bd -q30m
```

3.3 Forcing the queue

In some cases, you may find that the queue has gotten clogged. You can force a queue run by using the `-q` flag (with no value). It is entertaining to use the `-v` flag (verbose) when this is done to watch what happens:

```
/usr/lib/sendmail -q -v
```

3.4 Debugging

A fairly large number of debug flags are built into `sendmail`. Each debug flag has a number and a level, where higher levels mean to print out more information. The convention is that levels greater than 9 are absurd; that is, they print out so much information that you wouldn't normally want to see them except for debugging that particular piece of code. Debug flags are set with the `-d` option:

`-d` *debug-list*

with the syntax:

debug-list:	*debug-option* [, *debug-option*]
debug-option:	*debug-range* [. *debug-level*]
debug-range:	*integer* | *integer* – *integer*
debug-level:	*integer*

For example,

`-d12`	Set flag 12 to level 1.
`-d12.3`	Set flag 12 to level 3.
`-d3-17`	Set flags 3 through 17 to level 1.
`-d3-17.4`	Set flags 3 through 17 to level 4.

For a complete list of the available debug flags, look at the code (they are too dynamic to keep this documentation up to date).

3.5 Trying a different configuration file

An alternative configuration file can be specified by using the `-C` flag. For example,

```
/usr/lib/sendmail -Ctest.cf
```

uses the configuration file `test.cf` instead of the default `/usr/lib/sendmail.cf`. If the `-C` flag has no value, it defaults to `sendmail.cf` in the current directory.

3.6 Changing the values of options

Options can be overridden by using the `-o` flag. For example,

```
/usr/lib/sendmail -oT2m
```

sets the `T` (timeout) option to two minutes for this run only.

4. Tuning

You may want to change some of the configuration parameters, depending on the requirements of your site. Most of these are set by using an option in the configuration file. For example, the line `OT3d` sets option `T` to the value `3d` (3 days).

Most of these options default appropriately for most sites. However, sites having very high mail loads may find they need to tune them as appropriate for their mail load. In particular, sites experiencing a large number of small messages, many of which are delivered to many recipients, may find that they need to adjust the parameters dealing with queue priorities.

4.1 Timeouts

All time intervals are set by using a scaled syntax. For example, `10m` represents 10 minutes, whereas `2h30m` represents 2½ hours. The full set of scales is

`s` seconds

`m` minutes

`h` hours

`d` days

w weeks

4.1.1 Queue Interval

The argument to the `-q` flag specifies how often a subdaemon will run the queue. This is typically set to between 15 minutes and 1 hour.

4.1.2 Read timeouts

It is possible to time out when reading the standard input or when reading from a remote SMTP server. Technically, this is not acceptable within the published protocols. However, it might be appropriate to set it to something large in certain environments (such as an hour). This will reduce the chance of large numbers of idle daemons piling up on your system. This timeout is set by using the `r` option in the configuration file.

4.1.3 Message timeouts

After sitting in the queue for a few days, a message will time out. This is to insure that at least the sender is aware of the inability to send a message. The timeout is typically set to 3 days. This timeout is set by using the `T` option in the configuration file.

The time of submission is set in the queue, rather than the amount of time left until the timeout. As a result, you can flush messages that have been hanging for a short period by running the queue with a short message timeout. For example,

```
/usr/lib/sendmail -oT1d -q
```

will run the queue and flush anything that is 1 day old.

4.2 Forking during queue runs

By setting the `Y` option, `sendmail` will fork before each individual message while running the queue. This will prevent `sendmail` from consuming large amounts of memory, so it may be useful in memory-poor environments. However, if the `Y` option is not set, `sendmail` will keep track of hosts that are down during a queue run, which can improve performance dramatically.

4.3 Queue priorities

Every message is assigned a priority when it is first submitted, consisting of the message size (in bytes) offset by the message class times the *work class factor*, and the number of recipients times the

work recipient factor.

$$pri = size - (class * wrk) + (nrcpt * wrkrcpt)$$

The priority plus the creation time of the message (in seconds since January 1, 1970) is used to order the queue. Higher numbers for the priority mean that the message will be processed later when running the queue.

The message size is included so that large messages are penalized relative to small messages. The message class allows users to send high-priority messages by including a `Precedence:` field in their message; the value of this field is looked up in the `P` lines of the configuration file. Because the number of recipients affects the amount of load a message presents to the system, this is also included in the priority.

The recipient and class factors can be set in the configuration file by using the `y` and `z` options respectively. They default to 1000 (for the recipient factor) and 1800 (for the class factor). The initial priority is

$$pri = size - (class * z) + (nrcpt * y)$$

(Remember, higher values for this parameter actually mean that the job will be treated with lower priority.)

The priority of a job can also be adjusted each time it is processed (that is, each time an attempt is made to deliver it) by using the *work time factor*, set by the `Z` option. This is added to the priority, so it normally decreases the precedence of the job, on the grounds that jobs that have failed many times will tend to fail again in the future.

4.4 Load limiting

With the `x` option, `sendmail` can be asked to queue (but not deliver) mail if the system load average gets too high. When the load average exceeds the value of the `x` option, the delivery mode is set to `q` (queue only) if the *queue factor* (`q` option) divided by the difference in the current load average and the `x` option plus 1 exceeds the priority of the message. That is, the message is queued if

$$pri > \frac{QF}{LA - x + 1}$$

The `q` option defaults to 10000, so each point of load average is worth 10000 priority points (bytes + seconds + offsets).

For drastic cases, the `x` option defines a load average at which `sendmail` will refuse to accept network connections. Locally generated mail (including incoming UUCP mail) is still accepted.

4.5 Delivery mode

`sendmail` can operate in a number of delivery modes set by the `d` configuration option. These modes specify how quickly mail will be delivered. Legal modes are

`i` Deliver interactively (synchronously).

`b` Deliver in the background (asynchronously).

`q` Queue only (don't deliver).

There are trade-offs. Mode `i` passes the maximum amount of information to the sender, but is hardly ever necessary. Mode `q` puts the minimum load on your machine, but means that delivery may be delayed for up to the queue interval. Mode `b` is probably a good compromise. However, this mode can generate large numbers of processes if you have a mailer that takes a long time to deliver a message.

4.6 Log level

The level of logging can be set for `sendmail`. The default with a standard configuration table is level 9. The levels are as follows:

`0` No logging.

`1` Major problems only.

`2` Message collections and failed deliveries.

`3` Successful deliveries.

`4` Messages being deferred (because of a host being down, and so on).

5 Normal message queues.

6 Unusual but benign incidents (for example, trying to process a locked queue file).

9 Log internal queue ID to external message ID mappings. This can be useful for tracing a message as it travels between several hosts.

12 Several messages that are basically only of interest when debugging.

16 Verbose information regarding the queue.

4.7 File modes

Some files may have a number of modes. The modes depend on what functionality you want and the level of security you require.

4.7.1 To setuid or not to setuid

`sendmail` can safely be made setuid to `root`. At the point where it is about to execute (see `exec`(2)) a mailer, it checks to see if the UID is 0; if so, it resets the UID and GID to a default (set by the `u` and `g` options). (You can override this by setting the `S` flag for mailers that are trusted and must be called as `root`.) However, this will cause mail processing to be accounted (by using `sar`(1)) to `root` rather than to the user sending the mail.

4.7.2 Temporary file modes

The mode of all temporary files that `sendmail` creates is determined by the `F` option. Reasonable values for this option are 0600 and 0644. If the more permissive mode is selected, it will not be necessary to run `sendmail` as `root` at all (even when running the queue).

4.7.3 Should my alias database be writable?

At Berkeley the alias database (`/usr/lib/aliases*`) is mode 666. There are some dangers inherent in this approach: any user can add himself to any list or can steal any other user's mail.

The database that `sendmail` actually uses is represented by the files `aliases.dir` and `aliases.pag` (both in `/usr/lib`). The mode on these files should match the mode on `/usr/lib/aliases`. If `aliases` is writable and the `dbm` files (`aliases.dir` and `aliases.pag`) are not, users will be unable to

make their desired changes to the actual database. However, if `aliases` is read-only and the `dbm` files are writable, a slightly sophisticated user can arrange to steal mail anyway.

If your `dbm` files are not writable by the world or you do not have auto-rebuild enabled (with the `D` option), you must be careful to reconstruct the alias database each time you change the text version:

```
newaliases
```

If this step is ignored or forgotten, any intended changes will also be ignored or forgotten.

5. The configuration file

This section describes the configuration file in detail and gives hints on how to write one of your own if you have to.

The syntax of the configuration file is designed to be reasonably easy to parse, because this is done every time `sendmail` starts up, rather than easy for a human to read or write. On the "future project" list is a configuration-file compiler.

An overview of the configuration file is given first, followed by details of the semantics.

5.1 The syntax

The configuration file is organized as a series of lines, each of which begins with a single character defining the semantics for the rest of the line. Lines beginning with a space or a tab are continuation lines (although the semantics are not well defined in many places). Blank lines and lines beginning with a number symbol (#) are comments.

5.1.1 `R` and `S`: Rewriting rules

The core of address parsing is the rewriting rules. These are an ordered production system. `sendmail` scans through the set of rewriting rules looking for a match on the left side (*ls*) of the rule. When a rule matches, the address is replaced by the right side (*rs*) of the rule.

There are several sets of rewriting rules. Some of the rewriting sets are used internally and must have specific semantics. Other rewriting sets do not have specifically assigned semantics and may be referenced by the mailer definitions or by other rewriting sets.

The syntax of these two commands is as follows:

S*n*

sets the current set of rules being collected to *n*. If you begin a set more than once, it deletes the old definition.

R *ls rs comments*

These fields must be separated by at least one tab character; there may be embedded spaces in the fields. *ls* is a pattern that is applied to the input. If it matches, the input is rewritten to *rs*. The comments are ignored.

5.1.2 D: Define macro

Macros are named with a single character. These may be selected from the entire ASCII set, but user-defined macros should be selected from the set of uppercase letters only. Lowercase letters and special symbols are used internally.

The syntax for macro definitions is

D*xval*

where *x* is the name of the macro and *val* is the value it should have. Macros can be interpolated in most places by using the escape sequence $*x*.

5.1.3 C and F: Define classes

Classes of words can be defined to match on the left side of the rewriting rules. For example, a class of all local names for this site might be created so that attempts to send to oneself can be eliminated. These can be either defined directly in the configuration file or read in from another file. Classes can be given names from the set of uppercase letters. Lowercase letters and special characters are reserved for system use.

The syntax is

C*c word1 word2 ...*
F *c file*

The first form defines the class *c* to match any of the named words. It is permissible to split them among multiple lines; for example, the two

forms

```
CHmonet ucbmonet
```

and

```
CHmonet
CHucbmonet
```

are equivalent. The second form reads the elements of the class *c* from *file*.

5.1.4 M: Define mailer

Programs and interfaces to mailers are defined in this line. The syntax is

`M` *name*, {*field=value*}*

where *name* is the name of the mailer (used internally only) and the *field=value* pairs define attributes of the mailer. The fields are

`Path`	the pathname of the mailer
`Flags`	special flags for this mailer
`Sender`	a rewriting set for sender addresses
`Recipient`	a rewriting set for recipient addresses
`Argv`	an argument vector to pass to this mailer
`Eol`	the end-of-line string for this mailer
`Maxsize`	the maximum message length to this mailer

Only the first character of the field name is checked.

5.1.5 H: Define header

The format of the header lines that `sendmail` inserts into the message are defined by the `H` line. The syntax of this line is

`H[?` *mflags* `?]` *hname* `:` *htemplate*

Continuation lines are reflected directly into the outgoing message. *htemplate* is macro-expanded before insertion into the message. If *mflags* (surrounded by question marks) are specified, at least one of the specified flags must be stated in the mailer definition for this header to be automatically output. If one of these headers is in the input, it is reflected to the output regardless of these flags.

Some headers have special semantics that are described in the next section.

5.1.6 O: Set option

A number of random options can be set from a configuration file. Options are represented by single characters. The syntax of this line is

O *ovalue*

This sets option O to be *value*. Depending on the option, *value* may be a string, an integer, a Boolean (with legal values t, T, f, or F; the default is TRUE), or a time interval.

5.1.7 T: Define trusted users

Trusted users are those users who are permitted to override the sender address by using the -f flag. These typically are root, uucp, and network, but on some users it may be convenient to extend this list to include other users, perhaps to support a separate UUCP login for each host. The syntax of this line is

T *user1 user2 ...*

There may be more than one of these lines.

5.1.8 P: Precedence definitions

Values for the Precedence: field can be defined by using the P control line. The syntax of this field is

P *name=num*

When *name* is found in a Precedence: field, the message class is set to *num*. Higher numbers mean higher precedence. Numbers less than 0 have the special property that error messages will not be returned. The default precedence is 0. For example, at Berkeley the list of precedences is

```
Pfirst-class=0
Pspecial-delivery=100
Pjunk=-100
```

5.2 The semantics

This section describes the semantics of the configuration file.

5.2.1 Special macros and conditionals

Macros are interpolated by using the construct `$`*x*, where *x* is the name of the macro to be interpolated. In particular, lowercase letters are reserved to have special semantics used to pass information in or out of `sendmail`, and some special characters are reserved to provide conditionals, and so on.

The syntax of conditionals is

`$?`*x text1* `$|` *text2* `$.`

This interpolates *text1* if the macro `$`*x* is set, and *text2* otherwise. The `else` clause (`$|`) may be omitted.

The following macros must be defined to transmit information into `sendmail`:

`e`	the SMTP entry message
`j`	the "official" domain name for this site
`l`	the format of the `From` line
`n`	the name of the daemon (for error messages)
`o`	the set of "operators" in addresses
`q`	default format of sender address

The `$e` macro is printed out when SMTP starts up. The first word must be the `$j` macro; this should be in RFC 821 format. The `$l` and `$n` macros can be considered constants except under very unusual circumstances. The `$o` macro consists of a list of characters that will be considered tokens and that will separate tokens when doing parsing. For example, if `@` were in the `$o` macro, the input `a@b` would be scanned as three tokens: `a`, `@`, and `b`. Finally, the `$q` macro specifies how an address should appear in a message when it is defaulted. For example, at Berkeley these definitions are:

```
De$j sendmail $v ready at $b
DnMAILER-DAEMON
DlFrom $g  $d
Do.:%@!^=/
Dq$g$?x ($x)$.
Dj$H.$D
```

An acceptable alternative for the `$q` macro is `"$?x$x $.<$g>`.

These correspond to the following two formats:

```
eric@Berkeley (Eric Allman)
Eric Allman <eric@Berkeley>
```

Some macros are defined by `sendmail` for interpolation into `argv`'s for mailers or for other contexts. These macros are

`a`	the origination date in Arpanet format
`b`	the current date in Arpanet format
`c`	the hop count
`d`	the date in UNIX (`ctime`) format
`f`	the sender (from) address
`g`	the sender address relative to the recipient
`h`	the recipient host
`i`	the queue ID
`p`	`sendmail`'s PID
`r`	the protocol used
`s`	the sender's host name
`t`	a numeric representation of the current time
`u`	the recipient user
`v`	the version number of `sendmail`
`w`	the host name of this site
`x`	the full name of the sender
`z`	the home directory of the recipient

There are three types of dates that can be used. The `$a` and `$b` macros are in Arpanet format; `$a` is the time as extracted from the `Date:` line of the message (if there was one), and `$b` is the current date and time (used for postmarks). If no `Date:` line is found in the incoming message, `$a` is set to the current time. The `$d` macro is equivalent to the `$a` macro in UNIX (`ctime`) format.

The `$f` macro is the ID of the sender as originally determined; when mailing to a specific host, the `$g` macro is set to the address of the sender *relative to the recipient.* For example, if `eric` sends to `bollard@matisse` from the machine `ucbarpa`, the `$f` macro will be `eric` and the `$g` macro will be `eric@ucbarpa`.

The `$x` macro is set to the full name of the sender. This can be determined in several ways. The first way is to pass it as a flag to

`sendmail`. The second choice is the value of the `Full-name:` line in the header if it exists, and the third choice is the comment field of a `From:` line. If all of these fail, and if the message is being originated locally, the full name is looked up in the `/etc/passwd` file.

When sending, the `$h`, `$u`, and `$z` macros get set to the host, user, and home directory (if local) of the recipient. The first two are set from the `$@` and `$:` part of the rewriting rules, respectively.

The `$p` and `$t` macros are used to create unique strings (for example, for the `Message-Id:` field). The `$i` macro is set to the queue ID on this host; if put into the time stamp line, it can be extremely useful for tracking messages. The `$v` macro is set to be the version number of `sendmail`; this is normally put in time stamps and has been proven extremely useful for debugging. The `$w` macro is set to the name of this host if it can be determined. The `$c` field is set to the "hop count;" that is, the number of times this message has been processed. This can be determined by using the `-h` flag on the command line or by counting the time stamps in the message.

The `$r` and `$s` fields are set to the protocol used to communicate with `sendmail` and the sending host name; these are not supported in the current version.

5.2.2 Special classes

The class `$=w` is set to be the set of all names this host is known by. This can be used to delete local host names.

5.2.3 The left side

The left side of the rewriting rules contains a pattern. Normal words are simply matched directly. Metasyntax is introduced with a dollar sign. The metasymbols are:

`$*`	Match zero or more tokens.
`$+`	Match one or more tokens.
`$-`	Match exactly one token.
`$=`*x*	Match any token in class *x*.
`$~`*x*	Match any token not in class *x*.

If any of these match, they are assigned to the symbol `$`*n* for replacement on the right side, where *n* is the index in *ls*. For example, if *ls*

```
$-:$+
```

is applied to the input

```
UCBARPA:eric
```

the rule will match, and the values passed to *rs* will be

```
$1    UCBARPA
$2    eric
```

5.2.4 The right side

When the left side of a rewriting rule matches, the input is deleted and replaced by the right side. Tokens are copied directly from *rs* unless they begin with a dollar sign. Metasymbols are

`$`*n*	Substitute indefinite token *n* from *ls*.
`$[`*name*`$]`	Canonicalize *name*.
`$>`*n*	Call rule set *n*.
`$#`*mailer*	Resolve to *mailer*.
`$@`*host*	Specify *host*.
`$:`*user*	Specify *user*.

The `$`*n* syntax substitutes the corresponding value from a `$+`, `$-`, `$*`, `$=`, or `$~` match on *ls*. It may be used anywhere.

A host name enclosed between `$[` and `$]` is looked up by using the `gethostbyaddr`(3N) routines and replaced by the canonical name. For example, `$[csam$]` would become `lbl-csam.arpa` and `$[[128.32.130.2]$]` would become `vangogh.berkeley.edu`.

The `$>` *n* syntax causes the remainder of the line to be substituted as usual and then passed as the argument to rule set *n*. The final value of rule set *n* then becomes the substitution for this rule.

The `$#` syntax should be used only in rule set 0. It causes evaluation of the rule set to terminate immediately and signals to `sendmail` that the address has completely resolved. The complete syntax is

`$#`*mailer*`$@`*host*`$:`*user*

This specifies the {*mailer*, *host*, *user*} triple necessary to direct the mailer. If the mailer is local, the host part may be omitted. The mailer and host must be a single word, but the user may be multipart.

rs may also be preceded by a `$@` or a `$:` to control evaluation. A `$@` prefix causes the rule set to return with the remainder of *rs* as the value. A `$:` prefix causes the rule to terminate immediately but the rule set to continue; this can be used to avoid continued application of a rule. The prefix is stripped before continuing.

The `$@` and `$:` prefixes may precede a `$>`. For example,

```
R$+     $:$>7$1
```

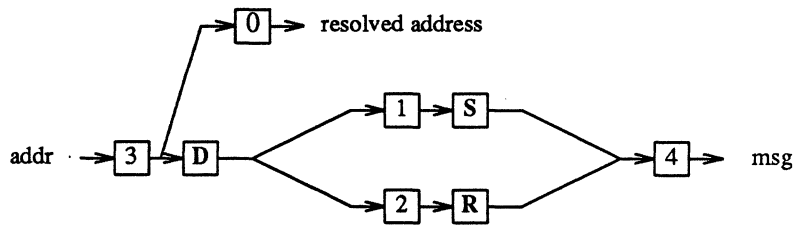
matches anything, passes that to rule set 7, and continues; the `$:` is necessary to avoid an infinite loop.

Substitution occurs in the order described; that is, parameters from *ls* are substituted, host names are canonicalized, subroutines are called, and finally `$#`, `$@`, and `$:` are processed.

5.2.5 Semantics of rewriting rule sets

There are five rewriting sets that have specific semantics, as depicted in Figure A-1.

Figure A-1. Rewriting set semantics



In the figure, D is the sender domain addition, S is the mailer-specific sender rewriting, and R is mailer-specific recipient rewriting.

Rule set 3 should turn the address into canonical form. This form should have the basic syntax

local-part`@`*host-domain-spec*

If no `@` is specified, the *host-domain-spec* may be appended from the sender address (if the `C` flag is set in the mailer definition corresponding to the `sending` mailer). Rule set 3 is applied by `sendmail` before doing anything with any address.

Rule set 0 is applied after rule set 3 to addresses that are going to specify recipients. It must resolve to a {*mailer*, *host*, *user*} triple. The mailer must be defined in the mailer definitions from the configuration file. The host is defined into the `$h` macro for use in the `argv` expansion of the specified mailer.

Rule sets 1 and 2 are applied to all sender and recipient addresses respectively. They are applied before any specification in the mailer definition. They must never resolve.

Rule set 4 is applied to all addresses in the message. It is typically used to translate internal to external form.

5.2.6 Mailer flags

A number of flags may be associated with each mailer, each identified by a letter of the alphabet. Many of them are assigned semantics internally. These are detailed in section 8. Any other flags may be used freely to conditionally assign headers to messages destined for particular mailers.

5.2.7 The `error` mailer

The mailer with the special name `error` can be used to generate a user error. The (optional) host field is a numeric exit status to be returned, and the user field is a message to be printed. For example, the entry

```
$#error$:Host unknown in this domain
```

on *rs* of a rule will cause the specified error to be generated if *ls* matches. This mailer is only functional in rule set 0.

5.3 Building a configuration file from scratch

Building a configuration table from scratch is an extremely difficult job. Fortunately, it is almost never necessary to do so; nearly every situation that may come up can be resolved by changing an existing

table. In any case, it is critical that you understand what it is that you are trying to do and come up with a philosophy for the configuration table. This section is intended to explain what the real purpose of a configuration table is and to give you some ideas for what your philosophy might be.

5.3.1 What you are trying to do

The configuration table has three main purposes. The first and simplest is to set up the environment for `sendmail`. This involves setting the options, defining a few critical macros, and so on. These are described in other places.

The second purpose is to rewrite addresses in the message. This should typically be done in two phases. The first phase maps addresses in any format into a canonical form. This should be done in rule set 3. The second phase maps this canonical form into the syntax appropriate for the receiving mailer. `sendmail` does this in three subphases. Rule sets 1 and 2 are applied to all sender and recipient addresses respectively. After this, you can specify per-mailer rule sets for both sender and recipient addresses; this allows mailer-specific customization. Finally, rule set 4 does any default conversion to external form.

The third purpose of the configuration table is to map addresses into the actual set of instructions necessary to get the message delivered. Rule set 0 must resolve to the internal form, which is in turn used as a pointer to a mailer descriptor. The mailer descriptor describes the interface requirements of the mailer.

5.3.2 Philosophy

The particular philosophy you choose will depend heavily on the size and structure of your organization.

One general point applies to all of the philosophies presented here: it is almost always a mistake to try to do full name resolution. For example, if you are trying to get names of the form `user@host` to the Arpanet, it does not pay to route them to

```
xyzvax!decvax!ucbvax!c70:user@host
```

because you then depend on several links not under your control. The

best approach to this problem is to simply forward to `xyzvax!user@host` and let `xyzvax` worry about it from there. In summary, just get the message closer to the destination, rather than determining the full path.

5.3.2.1 Large site, many hosts: Minimum information

Berkeley is an example of a large site, that is, more than two or three hosts and multiple mail connections. The only reasonable philosophy in this environment is to designate one host as the guru for the site. That host must be able to resolve any piece of mail it receives. The other sites should have the minimum amount of information they can get away with. In addition, any information they have should be hints rather than solid information.

For example, a typical site on the Berkeley local Ethernet is `monet`. When `monet` receives mail for delivery, it checks whether it knows that the destination host is directly reachable; if it is, mail is sent to that host. If it receives mail for an unknown host, it just passes it directly to `ucbvax`, the master host. `ucbvax` may determine that the host name is illegal and reject the message, or may be able to make the delivery. However, it is important to note that when a new mail connection is added, the only host that must have its tables updated is `ucbvax`; the others may be updated if convenient, but this is not critical.

This picture is slightly muddied because of network connections that are not actually located on `ucbvax`. For example, some UUCP connections are currently on `ucbarpa`. However, `monet` does not know about this; the information is hidden totally between `ucbvax` and `ucbarpa`. Mail going from `monet` to a UUCP host is transferred via the Ethernet from `monet` to `ucbvax`, then via the Ethernet from `ucbvax` to `ucbarpa`, and it is then is submitted to UUCP. Although this involves some extra hops, it is considered an acceptable trade-off.

An interesting point is that it would be possible to update `monet` to send appropriate UUCP mail directly to `ucbarpa` if the load got too high. If `monet` failed to note a host as connected to `ucbarpa`, it would go via `ucbvax` as before, and if `monet` incorrectly sent a message to `ucbarpa`. it would still be sent by `ucbarpa` to

ucbvax as before. The only problem that can occur is loops; for example, if ucbarpa thought that ucbvax had the UUCP connection and vice versa. For this reason, updates should always happen to the master host first.

This philosophy results as much from the need to have a single source for the configuration files (typically built using m4(1) or some similar tool) as any logical need. Maintaining more than three separate tables by hand is essentially an impossible job.

5.3.2.2 Small site: Complete information

A small site (two or three hosts and few external connections) may find it more reasonable to have complete information at each host. This would require that each host know exactly where each network connection is, possibly including the names of each host on that network. As long as the site remains small and the configuration remains relatively static, the update problem will probably not be too great.

5.3.2.3 Single host

This is in some sense the trivial case. The only big issue is trying to ensure that you don't have to know too much about your environment. For example, if you have a UUCP connection you might find it useful to know about the names of hosts connected directly to you, but this is really not necessary because it may be determined from the syntax.

5.3.3 Relevant Issues

The canonical form you use should almost certainly be as specified in the Arpanet protocols RFC 819 and RFC 822. Copies of these RFCs are included on the sendmail tape as doc/rfc819.lpr and doc/rfc822.lpr.

RFC 822 describes the format of the mail message itself. sendmail follows this RFC closely, to the extent that many of the standards described in this document can not be changed without changing the code. In particular, the following characters have special interpretations:

```
< > ( ) " \
```

Any attempt to use these characters for other than their RFC 822 purpose in addresses is probably doomed to disaster.

RFC 819 describes the specifics of the domain-based addressing. This is touched on in RFC 822 as well. Essentially, each host is given a name that is a right-to-left dot-qualified pseudo-path from a distinguished root. The elements of the path need not be physical hosts; the domain is logical rather than physical. For example, at Berkeley one legal host might be `a.CC.Berkeley.EDU`; reading from right to left, `EDU` is a top-level domain comprising educational institutions, `Berkeley` is a logical domain name, `CC` represents the Computer Center (in this case a strictly logical entity), and `a` is a host in the Computer Center.

Beware when reading RFC 819 that there are a number of errors in it.

5.3.4 How to proceed

Once you have decided on a philosophy, it is worth examining the available configuration tables to see if any of them are close enough to your needs to use parts of them. Even under the worst of conditions, there is a fair amount of boilerplate that can be collected safely.

The next step is to build rule set 3. This will be the hardest part of the job. Beware of doing too much to the address in this rule set, because anything you do will reflect through to the message. In particular, stripping of local domains is best deferred, because this can leave you with addresses with no domain spec at all. `sendmail` likes to append the sending domain to addresses with no domain, so this can change the semantics of addresses. Also try to avoid fully qualifying domains in this rule set. Although technically legal, this can lead to unpleasantly and unnecessarily long addresses reflected into messages. The Berkeley configuration files define rule set 9 to qualify domain names and strip local domains. This is called from rule set 0 to get all addresses into a cleaner form.

Once you have rule set 3 finished, the other rule sets should be relatively trivial. If you need hints, examine the supplied configuration tables.

5.3.5 Testing the rewriting rules: The `-bt` flag

When you build a configuration table, you can do a certain amount of testing by using the test mode of `sendmail`. For example, you could invoke `sendmail` as

```
sendmail -bt -Ctest.cf
```

which would read the configuration file `test.cf` and enter test mode. In this mode, you enter lines of the form

rwset address

where *rwset* is the rewriting set you want to use and *address* is an address to apply the set to. Test mode shows you the steps it takes as it proceeds, finally showing you the address it ends up with. You may use a comma-separated list of *rwset*s for sequential application of rules to an input; rule set 3 is always applied first. For example,

```
1,21,4 monet:bollard
```

first applies rule set 3 to the input `monet:bollard`. Rule set 1 is then applied to the output of rule set 3, followed similarly by rule sets 21 and 4.

If you need more detail, you can also use the `-d21` flag to turn on more debugging. For example,

```
sendmail -bt -d21.99
```

turns on an incredible amount of information; a single word address is probably going to print out several pages' worth of information.

5.3.6 Building mailer descriptions

To add an outgoing mailer to your mail system, you will have to define the characteristics of the mailer.

Each mailer must have an internal name. This can be arbitrary, except that the names `local` and `prog` must be defined.

The pathname of the mailer must be given in the `P` field. If this mailer should be accessed via an IPC connection, use the string `[IPC]` instead.

The `F` field defines the mailer flags. You should specify an `f` or `r` flag to pass the name of the sender as a `-f` or `-r` flag respectively. These flags are only passed if they were passed to `sendmail`, so that mailers that give errors under some circumstances can be placated. If the mailer is not picky, you can just specify `-f$g` in the `argv` template. If the mailer must be called as `root`, the `S` flag should be given; this will not reset the UID before calling the mailer.

> *Note:* `sendmail` must be running setuid to `root` for this to work.

If this mailer is local (that is, will perform final delivery rather than another network hop), the `-l` flag should be given. Quote characters (backslashes and double quotation marks) can be stripped from addresses if the `s` flag is specified; if this is not given, they are passed through. If the mailer is capable of sending to more than one user on the same host in a single transaction, the `m` flag should be stated. If this flag is on, the `argv` template containing `$u` will be repeated for each unique user on a given host. The `e` flag will mark the mailer as being expensive, which will cause `sendmail` to defer connection until a queue run.

> *Note:* The `c` configuration option must be given for this to be effective.

An unusual case is the `C` flag. This flag applies to the mailer that the message is received from, rather than the mailer being sent to; if set, the domain spec of the sender (that is, the `@host.domain` part) is saved and is appended to any addresses in the message that do not already contain a domain spec. For example, a message of the form

```
From: eric@ucbarpa
To: wnj@monet, mckusick
```

will be modified to

```
From: eric@ucbarpa
To: wnj@monet, mckusick@ucbarpa
```

if and only if the `C` flag is defined in the mailer corresponding to

`eric@ucbarpa`.

Other flags are described in section 8.

The `S` and `R` fields in the mailer description are per-mailer rewriting sets to be applied to sender and recipient addresses respectively. These are applied after the sending domain is appended and the general rewriting sets (numbers 1 and 2) are applied, but before the output rewrite (rule set 4) is applied. A typical use is to append the current domain to addresses that do not already have a domain. For example, a header of the form

```
From: eric
```

might be changed to be

```
From: eric@ucbarpa
```

or

```
From: ucbvax!eric
```

depending on the domain it is being shipped into. These sets can also be used to do special-purpose output rewriting in cooperation with rule set 4.

The `E` field defines the string to use as an end-of-line indication. A string containing only newline is the default. The usual backslash escapes (`\r`, `\n`, `\f`, `\b`) may be used.

Finally, an `argv` template is given as the `E` field. It may have embedded spaces. If there is no `argv` with a `$u` macro in it, `sendmail` will speak SMTP to the mailer. If the pathname for this mailer is `[IPC]`, the `argv` should be

```
IPC $h [ port ]
```

where *port* is the optional port number to connect to.

For example, the specifications

```
Mlocal, P=/bin/mail, F=rlsm S=10, R=20, A=mail -d $u
Mether, P=[IPC], F=meC, S=11, R=21, A=IPC $h, M=100000
```

specify a mailer to do local delivery and a mailer for Ethernet delivery.

The first is called `local`, is located in the file `/bin/mail`, takes a picky `-r` flag, and does local delivery. Quotes should be stripped from addresses, and multiple users can be delivered at once. Rule set 10 should be applied to sender addresses in the message and rule set 20 should be applied to recipient addresses. The `argv` to send to a message will be the word `mail`, the word `-d`, and words containing the name of the receiving user.

> *Note:* The A/UX implementation of `sendmail` does not support the `-d` option.

If a `-r` flag is inserted, it will be between the words `mail` and `-d`. The second mailer is called `ether` and should be connected to via an IPC connection. It can handle multiple users at once, connections should be deferred, and any domain from the sender address should be appended to any receiver name without a domain. Sender addresses should be processed by rule set 11 and recipient addresses by rule set 21. There is a 100,000-byte limit on messages passed through this mailer.

6. Command line flags

Arguments must be presented with flags before addresses. The flags are as follows:

`-f` *addr*
: The sender's machine address is *addr*. This flag is ignored unless the real user is listed as a "trusted user" or if *addr* contains an exclamation point (because of certain restrictions in UUCP).

`-r` *addr*
: An obsolete form of `-f`.

`-h` *cnt*
: Sets the hop count to *cnt*. This represents the number of times this message has been processed by `sendmail` (to the extent that it is supported by the underlying networks). *cnt* is incremented during processing, and if it reaches `MAXHOP` (currently 30), `sendmail` throws away the message with an

error.

`-F`*name*
Set the full name of this user to *name*.

`-n` Don't do aliasing or forwarding.

`-t` Read the header for `To:`, `Cc:`, and `Bcc:` lines, and send to everyone listed in those lists. The `Bcc:` line will be deleted before sending. Any addresses in the argument vector will be deleted from the send list.

`-b`*x* Set operation mode to *x*. Operation modes are

`m`	Deliver mail (default).
`a`	Run in Arpanet mode.
`s`	Speak SMTP on input side.
`d`	Run as a daemon.
`t`	Run in test mode.
`v`	Just verify addresses; don't collect or deliver.
`i`	Initialize the alias database.
`p`	Print the mail queue.
`z`	Freeze the configuration file.

The special processing for the Arpanet includes reading the `From:` line from the header to find the sender, printing Arpanet-style messages (preceded by three-digit reply codes for compatibility with the FTP protocol [Neigus73, Postel74, Postel77]), and ending lines of error messages with <CRLF>.

`-q`*time*
Try to process the queued mail. If the time is given, `sendmail` will run through the queue at the specified interval to deliver queued mail; otherwise, it only runs once.

`-C`*file*
Use a different configuration file. `sendmail` runs as the invoking user (rather than `root`) when this flag is specified.

`-d`*level*
Set debugging level.

`-o`*xvalue*

Set option *x* to the specified `value`. These options are described in section 7.

Some options may be specified as primitive flags (provided for compatibility with `delivermail`). These are the `e`, `i`, `m`, and `v` options. In addition, the `f` option may be specified as the `-s` flag.

7. Configuration options

The following options may be set by using the `-o` flag on the command line or the `O` line in the configuration file. Many of them cannot be specified unless the invoking user is trusted.

`A`*file* — Use *file* as the alias file. If no file is specified, use `aliases` in the current directory.

`a`*N* — If set, wait up to *N* minutes for an `@:@` entry to exist in the alias database before starting up. If it does not appear in *N* minutes, rebuild the database (if the `D` option is also set) or issue a warning.

`B`*c* — Set the blank substitution character to *c*. Unquoted spaces in addresses are replaced by this character.

`c` — If an outgoing mailer is marked as being expensive, don't connect immediately. This requires that queueing be compiled in, because it will depend on a queue-run process to actually send the mail.

`d`*x* — Deliver in mode *x*. Legal modes are

- `i` Deliver interactively (synchronously).
- `b` Deliver in the background (asynchronously).
- `q` Just queue the message (deliver during queue run).

`D` — If set, rebuild the alias database if necessary and possible. If this option is not set, `sendmail` will never rebuild the alias database unless explicitly requested with `-bi`.

`e`*x* — Dispose of errors by using mode *x*. The values for *x* are

- `p` Print error messages (default).

	`q` No messages; just give exit status. `m` Mail back errors. `w` Write back errors (mail if user is not logged in). `e` Mail back errors and give zero exit status always.
`F`*n*	The temporary file mode, in octal. 644 and 600 are good choices.
`f`	Save `From` lines at the front of headers. Normally they are assumed redundant and discarded.
`g`*n*	Set the default GID for mailers to run in to *n*.
`H`*file*	Specify the help file for SMTP.
`i`	Ignore dots in incoming messages.
`L`*n*	Set the default log level to *n* .
`M`*xvalue*	Set the macro *x* to *value*. This is intended only for use from the command line.
`m`	Send to me too, even if I am in an alias expansion.
`N`*net-name*	The name of the home network; `ARPA` by default. The argument of an SMTP `HELO` command is checked against *host-name* . *net-name* where *host-name* is requested from the kernel for the current connection. If they do not match, `Received:` lines are augmented by the name that is determined in this manner so that messages can be traced accurately.
`o`	Assume that the headers may be in old format; that is, with spaces delimitting names. This turns on an adaptive algorithm: if any recipient address contains a comma, parenthesis, or angle bracket, it will be assumed that commas already exist. If this flag is not on, only commas delimit names. Headers are always output with commas between the names.
`Q`*dir*	Use *dir* as the queue directory.
`q`*factor*	Use *factor* as the multiplier in the map function to decide when to queue jobs rather than run them. This value is

divided by the difference between the current load average and the load average limit `x` flag) to determine the maximum message priority that will be sent. Defaults to 10000.

`r`*time* Read timeout after *time* interval.

`S`*file* Log statistics in *file*.

`s` Be super-safe when running things; that is, always instantiate the queue file, even if you are going to attempt immediate delivery. `sendmail` always instantiates the queue file before returning control to the client under any circumstances.

`T`*time* Set the queue timeout to *time*. After this interval, messages that have not been successfully sent will be returned to the sender.

`t`*S*`,`*D* Set the local time zone name to *S* for standard time and *D* for daylight time; this is only used under version 6.

`u`*n* Set the default UID for mailers to *n*. Mailers without the `S` flag in the mailer definition will run as this user.

`v` Run in verbose mode.

`x`*LA* When the system load average exceeds *LA*, just queue messages rather than send them.

`X`*LA* When the system load average exceeds *LA*, refuse incoming SMTP connections.

`y`*fact* Add *fact* to the priority (thus *lowering* the priority of the job) for each recipient. This value penalizes jobs with large numbers of recipients.

`Y` If set, deliver each job that is run from the queue in a separate process. Use this option if you are short of memory, because the default tends to consume considerable amounts of memory while the queue is being processed.

z*fact* Multiply *fact* by the message class (determined by the `Precedence:` field in the user header and the `P` lines in the configuration file) and subtract from the priority. Thus, messages with a higher priority will be favored.

Z*fact* Add *fact* to the priority every time a job is processed. Thus, each time a job is processed, its priority will be decreased by the indicated value. In most environments this should be positive, because hosts that are down are all too often down for a long time.

8. Mailer flags

The following flags may be set in the mailer description:

`f` The mailer wants a `-f` from flag, but only if this is a network forward operation (that is, the mailer will give an error if the executing user does not have special permissions).

`r` Same as `f`, but sends a `-r` flag.

`S` Don't reset the UID before calling the mailer. This would be used in a secure environment where `sendmail` ran as `root`. This could be used to avoid forged addresses. This flag is suppressed if given from an unsafe environment (for example, a user's `mail.cf` file).

`n` Do not insert a `From` line on the front of the message.

`l` This mailer is local (that is, final delivery will be performed).

`s` Strip quote characters off of the address before calling the mailer.

`m` This mailer can send to multiple users on the same host in one transaction. When a `$u` macro occurs in the `argv` part of the mailer definition, that field will be repeated as necessary for all qualifying users.

`F` This mailer wants a `From:` header line.

`D` This mailer wants a `Date:` header line.

M This mailer wants a `Message-Id:` header line.

x This mailer wants a `Full-Name:` header line.

P This mailer wants a `Return-Path:` line.

u Uppercase should be preserved in user names for this mailer.

h Uppercase should be preserved in host names for this mailer.

A This is an Arpanet-compatible mailer, and all appropriate modes should be set.

U This mailer wants `From` lines with the UUCP-style "`remote from` *host*" on the end.

e This mailer is expensive to connect to, so try to avoid connecting normally; any necessary connection will occur during a queue run.

X This mailer wants to use the hidden-dot algorithm as specified in RFC 821; basically, any line beginning with a dot will have an extra dot prepended (to be stripped at the other end). This ensures that lines in the message containing a dot will not terminate the message prematurely.

L Limit the line lengths as specified in RFC 821.

P Use the return path in the SMTP `MAIL FROM:` command rather than just the return address. Although this is required in RFC 821, many hosts do not process return paths properly.

I This mailer will be speaking SMTP to another `sendmail`; as such, it can use special protocol features. This option is not required (that is, if this option is omitted the transmission will still operate successfully, although perhaps not as efficiently as possible).

C If mail is received from a mailer with this flag set, any addresses in the header that do not have an at sign (`@`) after being rewritten by rule set 3 will have the `@domain` clause from the sender tacked on. This allows mail with headers of the form

```
From: usera@hosta
To: userb@hostb, userc
```

to be rewritten as:

```
From: usera@hosta
To: userb@hostb, userc@hosta
```

automatically.

`E` Escape lines beginning with `From` in the message with a `>`.

9. Other configurations

Some configuration changes can be made by recompiling `sendmail`. These are located in three places:

`md/config.m4`
: These contain operating system-dependent descriptions. They are interpolated into the makefiles in the `src` and `aux` directories. This includes information about what version of the UNIX system you are running, what libraries you have to include, and so on.

`src/conf.h`
: Configuration parameters that may be tweaked by the installer are included in `conf.h`.

`src/conf.c`
: Some special routines and a few variables may be defined in `conf.c`. For the most part, these are selected from the settings in `conf.h`.

9.1 Parameters in `md/config.m4`

The following compilation flags may be defined in the `m4CONFIG` macro in `md/config.m4` to define the environment in which you are operating.

`V6` If set, this will compile a version 6 system, with 8-bit UIDs, single character TTY IDs, and so on.

`VMUNIX`
: If set, you will be assumed to have a Berkeley 4BSD or 4.1BSD, including the `vfork`(2) system call, special types defined in

<sys/types.h> (for example, u_char), and so on.

If none of these flags are set, a version 7 system is assumed.

You will also have to specify what libraries to link with sendmail in the m4LIBS macro. Most notably, you will have to include -l*jobs* if you are running a 4.1BSD system.

9.2 Parameters in src/conf.h

Parameters and compilation options are defined in conf.h. Most of these need not normally be tweaked; common parameters are all in sendmail.cf. However, the sizes of certain primitive vectors, and so on, are included in this file. The numbers following the parameters are their default value.

MAXLINE [1024]
: The maximum line length of any input line. If message lines exceed this length, they will still be processed correctly; however, header lines, configuration file lines, alias lines, and so on, must fit within this limit.

MAXNAME [256]
: The maximum length of any name, such as a host or a user name.

MAXFIELD [2500]
: The maximum total length of any header field, including continuation lines.

MAXPV [40]
: The maximum number of parameters to any mailer. This limits the number of recipients that may be passed in one transaction.

MAXHOP [17]
: When a message has been processed more than this number of times, sendmail rejects the message on the assumption that there has been an aliasing loop. This can be determined from the -h flag or by counting the number of trace fields (that is, Received: lines) in the message header.

MAXATOM [100]
: The maximum number of atoms (tokens) in a single address. For

example, the address `eric@Berkeley` is three atoms.

`MAXMAILERS` [25]
The maximum number of mailers that may be defined in the configuration file.

`MAXRWSETS` [30]
The maximum number of rewriting sets that may be defined.

`MAXPRIORITIES` [25]
The maximum number of values for the `Precedence:` field that may be defined (by using the `P` line in `sendmail.cf`).

`MAXTRUST` [30]
The maximum number of trusted users that may be defined (by using the `T` line in `sendmail.cf`).

`MAXUSERENVIRON` [40]
The maximum number of items in the user environment that will be passed to subordinate mailers.

`QUEUESIZE` [600]
The maximum number of entries that will be processed in a single queue run.

Other compilation options specify whether or not specific code should be compiled in:

`DBM` If set, the `dbm` package is used (see `dbm(3X)` in *A/UX Programmer's Reference*). If not set, a much less efficient algorithm for processing aliases is used.

`NDBM`
If set, the new version of the `dbm` library that allows multiple databases will be used. `dbm` must also be set.

`DEBUG`
If set, debugging information is compiled in. To get the debugging output, the `-d` flag must be used.

`LOG`
If set, the `syslog` routine in use at some sites is used. This makes an informational log record for each message processed

and makes a higher priority log record for internal system errors.

`QUEUE`
: This flag should be set to compile in the queueing code. If this is not set, mailers must accept the mail immediately or it will be returned to the sender.

`SMTP`
: If set, the code to handle user and server SMTP will be compiled in. This is only necessary if your machine has some mailer that speaks SMTP.

`DAEMON`
: If set, code to run a daemon is compiled in. This code is for 4.2 or 4.3BSD.

`UGLYUUCP`
: If you have a UUCP host adjacent to you that is not running a reasonable version of `rmail`, you will have to set this flag to include the "`remote from` *sysname*" information on the `From` line. Otherwise, UUCP gets confused about where the mail came from.

`NOTUNIX`
: If you are not using a UNIX mail format, you can set this flag to turn off special processing of UNIX `From` lines.

9.3 Configuration in `src/conf.c`

Not all header semantics are defined in the configuration file. Header lines that should be included only by certain mailers (as well as other more obscure semantics) must be specified in the `HdrInfo` table in `conf.c`. This table contains the header name (which should be in all lowercase) and a set of header control flags. The flags are

`H_ACHECK`
: Normally when the check is made to see if a header line is compatible with a mailer, `sendmail` will not delete an existing line. If this flag is set, `sendmail` will delete even existing header lines. That is, if this bit is set and the mailer does not have flag bits set that intersect with the required mailer flags in the header definition in `sendmail.cf`, the header line is

always deleted.

H_EOH
: If this header field is set, treat it like a blank line; that is, it will signal the end of the header and the beginning of the message text.

H_FORCE
: Add this header entry even if one existed in the message before. If a header entry does not have this bit set, `sendmail` will not add another header line if a header line of this name already existed. This would normally be used to stamp the message by everyone who handled it.

H_TRACE
: If set, this is a time stamp (trace) field. If the number of trace fields in a message exceeds a preset amount, the message is returned on the assumption that it has an aliasing loop.

H_RCPT
: If set, this field contains recipient addresses. This is used by the `-t` flag to determine who to send to when it is collecting recipients from the message.

H_FROM
: This flag indicates that this field specifies a sender. The order of these fields in the `HdrInfo` table specifies `sendmail`'s preference for which field to return error messages to.

Now look at a sample `HdrInfo` specification:

```
struct hdrinfo   HdrInfo[] =
{
        /* originator fields, most to least significant  */
   "resent-sender",   H_FROM,
   "resent-from",   H_FROM,
   "sender",   H_FROM,
   "from",   H_FROM,
   "full-name",   H_ACHECK,
        /* destination fields */
   "to",   H_RCPT,
   "resent-to",   H_RCPT,
   "cc",   H_RCPT,
        /* message identification and control */
   "message",   H_EOH,
   "text",   H_ECH,
        /* trace fields */
   "received",   H_TRACE|H_FORCE,

   NULL,   0,
};
```

This structure indicates that the `To:`, `Resent-To:`, and `Cc:` fields all specify recipient addresses. Any `Full-Name:` field will be deleted unless the required mailer flag (indicated in the configuration file) is specified. The `Message:` and `Text:` fields will terminate the header; these are specified in new protocols (NBS80) or used by random dissenters around the network world. The `Received:` field will always be added and can be used to trace messages.

There are a couple of important points here. First, header fields are not added automatically just because they are in the `HdrInfo` structure; they must be specified in the configuration file to be added to the message. Any header fields mentioned in the configuration file but not mentioned in the `HdrInfo` structure have default processing performed; that is, they are added unless they were in the message already. Second, the `HdrInfo` structure only specifies cliched processing; certain headers are processed specially by ad hoc code regardless of the status specified in `HdrInfo`. For example, the `Sender:` and `From:` fields are always scanned on Arpanet mail to

determine the sender; this is used to perform the return-to-sender function. The `From:` and `Full-Name:` fields are used to determine the full name of the sender if possible; this is stored in the macro `$x` and used in a number of ways.

The file `conf.c` also contains the specification of Arpanet reply codes. These fall into four classifications

```
char  Arpa_Info[]    =   "050";   /* arbitrary info */
char  Arpa_TSyserr[] =   "455";   /* some (transient)
                                     system error */
char  Arpa_PSyserr[] =   "554";   /* some (permanent)
                                     system error */
char  Arpa_Usrerr[]  =   "554";   /* some (fatal) user
                                     error */
```

The class `Arpa_Info` is for any information that is not required by the protocol, such as forwarding information. `Arpa_TSyserr` and `Arpa_PSyserr` are printed by the `syserr` routine. `TSyserr` is printed out for transient errors, that is, errors that are likely to go away without explicit action on the part of a system administrator. `PSyserr` is printed for permanent errors. The distinction made is based on the value of `errno`. Finally, `Arpa_Usrerr` is the result of a user error and is generated by the `usrerr` routine. These are generated when the user has specified something wrong, and hence the error is permanent; that is, it will not work simply by resubmitting the request.

If it is necessary to restrict mail through a relay, the `checkcompat` routine can be modified. This routine is called for every recipient address. It can return `TRUE` to indicate that the address is acceptable and mail processing will continue, or it can return `FALSE` to reject the recipient. If it returns `FALSE`, it is up to `checkcompat` to print an error message (by using `usrerr`) saying why the message is rejected. For example, `checkcompat` could read

```
bool
checkcompat(to)
   register ADDRESS *to;
{
   if (MsgSize > 50000 && to->q_mailer != LocalMailer)
   {
      usrerr("Message too large for non-local delivery");
      NoReturn = TRUE;
      return (FALSE);
   }
   return (TRUE);
}
```

This would reject messages greater than 50000 bytes unless they were local. The `NoReturn` flag can be sent to suppress the return of the actual body of the message in the error return. The use of this routine is highly dependent on the implementation, and should be limited.

9.4 Configuration in `src/daemon.c`

The file `src/daemon.c` contains routines that are dependent on the local networking environment. The version supplied is specific to 4.3 BSD.

The routine `maphostname` is called to convert strings within `$[...$]` symbols. It can be modified to provide a more sophisticated service; for example, mapping UUCP host names to full paths.

10. Summary of support files

This is a summary of the support files that `sendmail` creates or generates.

`/usr/lib/sendmail`
: The binary of `sendmail`.

`/usr/bin/newaliases`
: A link to `/usr/lib/sendmail`; causes the alias database to be rebuilt. Running this program is equivalent to giving `sendmail` the `-bi` flag.

`/usr/bin/mailq`
Prints a listing of the mail queue. This program is equivalent to using the `-bp` flag to `sendmail`.

`/usr/lib/sendmail.cf`
The configuration file, in textual form.

`/usr/lib/sendmail.fc`
The configuration file represented as a memory image.

`/usr/lib/sendmail.hf`
The SMTP help file.

`/usr/lib/sendmail.st`
A statistics file; need not be present.

`/usr/lib/aliases`
The textual version of the alias file.

`/usr/lib/aliases.{pag,dir}`
The alias file in `dbm`(3X) format.

`/usr/spool/mqueue`
The directory in which the mail queue and temporary files reside.

`/usr/spool/mqueue/qf*`
Control (queue) files for messages.

`/usr/spool/mqueue/df*`
Data files.

`/usr/spool/mqueue/lf*`
Lock files.

`/usr/spool/mqueue/tf*`
Temporary versions of the `qf` files; used during queue file rebuild.

`/usr/spool/mqueue/nf*`
A file used when creating a unique ID.

`/usr/spool/mqueue/xf*`
A transcript of the current session.

Appendix B
Name Server Operations Guide for BIND

Contents

1. Introduction 2

2. Building a system with a name server 2
   2.1 Resolver routines in `libc` 2
   2.2 The name server 3

3. Types of server 4
   3.1 Master server 4
      3.1.1 Primary 4
      3.1.2 Secondary 4
   3.2 Caching-only server 4
   3.3 Remote server 5
   3.4 Slave server 5

4. Setting up your own domain 5
   4.1 DARPA Internet 6
   4.2 CSNET 6
   4.3 BITNET 6

5. Files 6
   5.1 Boot file 6
      5.1.1 Domain 7
      5.1.2 Directory 7
      5.1.3 Primary master 7
      5.1.4 Secondary master 8
      5.1.5 Caching-only server 8
      5.1.6 Forwarders 8
      5.1.7 Slave mode 9
      5.1.8 Remote server 9
   5.2 Cache initialization 9
      5.2.1 `root.cache` 9

5.3 Domain data files 10
5.3.1 named.local 10
5.3.2 hosts 10
5.3.3 hosts.rev 10
5.4 Standard Resource Record Format 10
5.4.1 $INCLUDE 12
5.4.2 $ORIGIN 12
5.4.3 SOA: start of authority 12
5.4.4 NS: Name server 13
5.4.5 A: address 13
5.4.6 HINFO: host information 13
5.4.7 WKS: well-known services 14
5.4.8 CNAME: canonical name 14
5.4.9 PTR: domain name pointer 15
5.4.10 MB: Mailbox 15
5.4.11 MR: Mail rename name 15
5.4.12 MINFO: Mailbox information 15
5.4.13 MG: Mail group member 16
5.4.14 MX: Mail exchanger 16
5.5 Sample files 17
5.5.1 Boot file 17
5.5.1.1 Primary master server 17
5.5.2 Secondary master server 17
5.5.2.1 Caching-only server 18
5.5.3 Remote server 19
5.5.3.1 /etc/resolv.conf 19
5.5.4 root.cache 19
5.5.5 named.local 19
5.5.6 hosts 21
5.5.7 host.rev 23

6. Domain management 23
6.1 /etc/rc.local 23
6.2 /etc/named.pid 23
6.3 /etc/hosts 24
6.4 Signals 24
6.4.1 Reload 24
6.4.2 Debugging 24

Appendix B

Name Server Operations Guide for BIND

This appendix was originally printed as the *Name Server Operations Guide for BIND Release 4.8* written by Kevin J. Dunlap and Michael J. Karels of the Computer Systems Research Group at the University of California at Berkeley, and is reproduced here for your reference with permission.

1. Introduction

The Berkeley Internet Name Domain (BIND) server implements the DARPA Internet name server for the UNIX operating system. A name server is a network service that enables clients to name resources or objects and share this information with other objects in the network. This in effect is a distributed data base system for objects in a computer network. BIND is fully intergrated into 4.3BSD network programs for use in storing and retrieving host names and address. The system administrator can configure the system to use BIND as a replacement to the original host table lookup of information in the network hosts file `/etc/hosts`. The default configuration for 4.3BSD uses BIND.

2. Building a system with a name server

BIND is comprised of two parts. One is the user interface called the `resolver` which consists of a group of routines that reside in the C library `/lib/libc.a`. Second is the actual server called `named`. This is a daemon that runs in the background and services queries on a given network port. The standard port for UDP and TCP is specified in `/etc/services`.

2.1 Resolver routines in `libc`

When building your 4.3BSD system you may either build the C library to use the name server resolver routines or use the host table lookup routines to do host name and address resolution. The default resolver for 4.3BSD uses the name server.

Building the C library to use the name server changes the way `gethostbyname` (3N), `gethostbyaddr` (3N), and `sethostent` (3N) do their functions. The name server renders `gethostent` (3N) obsolete, since it has no concept of a next line in the database. These library calls are built with the resolver routines needed to query the name server.

The `resolver` is comprised of a few routines that build query packets and exchange them with the name server.

Before building the C library, set the variable `HOSTLOOKUP` equal to `named` in `/usr/src/lib/libc/Makefile`. You then make and install the C library and compiler and then compile the rest of the 4.3BSD system. For more information see section 6.6 of "Installing and Operating 4.3BSD on the VAX".

2.2 The name server

The basic function of the name server is to provide information about network objects by answering queries. The specifications for this name server are defined in RFC882, RFC883, RFC973 and RFC974. These documents can be found in `/usr/src/etc/named/doc` in 4.3BSD or `ftp`ed from `sri-nic.arpa`. It is also recommeded that you read the related manual pages, `named`(1M), `resolver`(3N), and `resolver`(4).

The advantage of using a name server over the host table lookup for host name resolution is to avoid the need for a single centralized clearinghouse for all names. The authority for this information can be delegated to the different organizations on the network responsible for it.

The host table lookup routines require that the master file for the entire network be maintained at a central location by a few people. This works fine for small networks where there are only a few machines and the different organizations responsible for them cooperate. But this does not work well for large networks where machines cross organizational boundaries.

With the name server, the network can be broken into a hierarchy of domains. The name space is organized as a tree according to organizational or administrative boundaries. Each node, called a `domain`, is given a label, and the name of the domain is the concatenation of all the labels of the domains from the root to the current domain, listed from right to left separated by dots. A label need only be unique within its domain. The whole space is partitioned into several areas called `zones`, each starting at a domain and extending down to the leaf domains or to domains where other zones start. Zones

usually represent administrative boundaries. An example of a host address for a host at the University of California, Berkeley would look as follows:

```
monet.Berkeley.EDU
```

The top level domain for educational organizations is EDU; Berkeley is a subdomain of EDU and monet is the name of the host.

3. Types of server

There are several types of server: master, caching, remote, and slave.

3.1 Master server

A **master server** for a domain is the authority for that domain. This server maintains all the data corresponding to its domain. Each domain should have at least two master servers, a primary master and some secondary masters to provide backup service if the primary is unavailable or overloaded. A server may be a master for multiple domains, being primary for some domains and secondary for others.

3.1.1 Primary

A **primary master server** is a server that loads its data from a file on disk. This server may also delegate authority to other servers in its domain.

3.1.2 Secondary

A **secondary master server** is a server that is delegated authority and receives its data for a domain from a primary master server. At boot time, the secondary server requests all the data for the given zone from the primary master server. This server then periodically checks with the primary server to see if it needs to update its data.

3.2 Caching-only server

All servers are caching servers. This means that the server caches the information that it receives for use until the data expires. A **caching-only server** is a server that is not authoritative for any domain. This server services queries and asks other servers, who have the authority, for the information needed. All servers keep data in their cache until the data expires, based on a time to live field attached to the data when it is received from another server.

3.3 Remote server

A **remote server** is an option given to people who would like to use a name server on their workstation or on a machine that has a limited amount of memory and CPU cycles. With this option you can run all of the networking programs that use the name server without the name server running on the local machine. All of the queries are serviced by a name server that is running on another machine on the network.

3.4 Slave server

A **slave server** is a server that always forwards queries it cannot satisfy locally to a fixed list of forwarding servers instead of interacting with the master name servers for the root and other domains. The queries to the forwarding servers are recursive queries. There may be one or more forwarding servers, and they are tried in turn until the list is exhausted. A slave and forwarder configuration is typically used when you do not wish all the servers at a give site to be interacting with the rest of the Internet servers. A typically scenario would involve a number of workstations and a departmental timesharing machine with Internet access. The workstations might be administratively prohibited from having Internet access. To give the workstations the appearance of access to the Internet domain system, the workstations could be slave servers to the timesharing machine which would forward the queries and interact with other name servers to resolv the query before returning the answer. An added benefit of using the forwarding feature is that the central machine develops a much more complete cache of information that all the workstations can take advantage of. The use slave mode and forwarding is discussed further under the description of the named boot file commands.

4. Setting up your own domain

When setting up a domain that is going to be on a public network, the site administrator should contact the organization in charge of the network and request the appropriate domain registration form. An organization that belongs to multiple networks (such as CSNET, DARPA Internet, and BITNET) should register with only one network.

The contacts are as follows:

4.1 DARPA Internet

Sites that are already on the DARPA Internet and need information on setting up a domain should contact HOSTMASTER@SRI-NIC.ARPA. You may also want to be placed on the BIND mailing list, which is a mail group for people on the DARPA Internet running BIND. The group discusses future design decisions, operational problems, and other related topics. The address to request being placed on this mailing list is:

```
bind-request@ucbarpa.Berkeley.EDU.
```

4.2 CSNET

A CSNET member organization that has not registered its domain name should contact the CSNET Coordination and Information Center (CIC) for an application and information about setting up a domain.

An organization that already has a registered domain name should keep the CIC informed about how it would like its mail routed. In general, the CSNET relay will prefer to send mail via CSNET (as opposed to BITNET or the Internet) if possible. For an organization on multiple networks, this may not always be the preferred behavior. The `CIC` can be reached via electronic mail at `cic@sh.cs.net`, or by phone at (617) 497-2777.

4.3 BITNET

If you are on the BITNET and need to set up a domain, contact `INFO@BITNIC`.

5. Files

The name server uses several files to load its database. This section covers the files and their formats needed for `named`.

5.1 Boot file

The boot file is the file that is first read when `named` starts up. This tells the server what type of server it is, which zones it has authority over, and where to get its initial data. The default location for this file is `/etc/named.boot`. However this can be changed by setting the `BOOTFILE` variable when you compile `named` or by specifying the location on the command line when `named` is started up.

5.1.1 Domain

A default domain may be specified for the name server using a line such as

```
domain    Berkeley-EDU
```

The name server uses this information when it receives a query for a name without a "." that is not known. When it receives one of these queries, it appends the name in the second field to the query name. This is an obsolete facility which will be removed from future releases.

5.1.2 Directory

The directory line specifies the directory in which the name server should run, allowing the other filenames in the boot file to use relative pathnames.

```
directory      usr/local/domain
```

If you have more than a couple of named files to be maintained, you may wish to place the named files in a directory such as `/usr/local/domain` and adjust the directory command properly. The main purposes of this command are to make sure `named` is in the proper directory when trying to include files by relative pathnames with `$INCLUDE` and to allow named to run in a location that is reasonable to dump core if it feels the urge.

5.1.3 Primary master

The line in the boot file that designates the server as a primary server for a zone looks as follows:

```
primary      Berkeley-EDU ucbhosts
```

The first field specifies that the server is a primary one for the zone stated in the second field. The third field is the name of the file from which the data is read.

5.1.4 Secondary master

The line for a secondary server is similar to the primary except that it lists addresses of other servers (usually primary servers) from which the zone data will be obtained.

```
secondary       Berkeley-EDU   128320101283204 ucbhosts.bak
```

The first field specifies that the server is a secondary master server for the zone stated in the second field. The two network addresses specify the name servers that are primary for the zone. The secondary server gets its data across the network from the listed servers. Each server is tried in the order listed until it successfully receives the data from a listed server. If a filename is present after the list of primary servers, data for the zone will be dumped into that file as a backup. When the server is first started, the data are loaded from the backup file if possible, and a primary server is then consulted to check that the zone is still up-to-date.

5.1.5 Caching-only server

You do not need a special line to designate that a server is a caching server. What denotes a caching-only server is the absence of authority lines, such as `secondary` or `primary` in the boot file.

All servers should have a line as follows in the boot file to prime the name server, cache:

```
cache           root-cache
```

All cache files listed will be read in at named boot time and any values still valid will be reinstated in the cache and the root name server information in the cache files will always be used. For information on cache file see section "Cache Initialization."

5.1.6 Forwarders

Any server can make use of forwarder. A **forwarder** is another server capable of processing recursive queries that is willing to try resolving queries on behalf of other systems. The forwarders command specifies forwarders by internet address as follows:

```
forwarders          128320101283204
```

There are two main reasons for wanting to do so. First, the other systems may not have full network access and may be prevented from sending any IP packets into the rest of the network and therefore must rely on a forwarder that does have access to the full net. The second reason is that the forwarder sees a union of all queries as they pass through his server and therefore he builds up a very rich cache of data compared to the cache in a typical workstation name server. In effect, the forwarder becomes a meta-cache that all hosts can benefit from, thereby reducing the total number of queries from that site to the rest of the net.

5.1.7 Slave mode

Slave mode is used if the use of forwarders is the only possible way to resolve queries due to the lack of full net access or to prevent the name server from using other than the listed forwarders. Slave mode is activated by placing the simple command

```
slave
```

in the boot file. If `slave` is used, then you must specify forwarders. When in slave mode, the server will forward each query to each of the the forwarders until an answer is found or the list of forwarders is exhausted.

5.1.8 Remote server

To set up a host that will use a remote server instead of a local server to answer queries, the file `/etc/resolv.conf` needs to be created. This file designates the name servers on the network that should be sent queries. It is not advisable to create this file if you have a local server running. If this file exists it is read almost every time `gethostbyname()` or `gethostbyaddr()` is called.

5.2 Cache Initialization

5.2.1 `root.cache`

The name server needs to know the servers that are the authoritative name servers for the root domain of the network. To do this we have to prime the name server's cache with the addresses of these higher

authorities. The location of this file is specified in the boot file. This file uses the Standard Resource Record Format (aka. Masterfile Format) covered further on in this paper.

5.3 Domain data files

There are three standard files for specifying the data for a domain. These are `named.local`, `hosts` and `host.rev`. These files use the Standard Resource Record Format covered later in this paper.

5.3.1 named.local

This file specifies the address for the local loopback interface, better known as `localhost` with the network address `127.0.0.1`. The location of this file is specified in the boot file.

5.3.2 hosts

This file contains all the data about the machines in this zone. The location of this file is specified in the boot file.

5.3.3 hosts.rev

This file specifies the IN-ADDR.ARPA domain. This is a special domain for allowing address to name mapping. As internet host addresses do not fall within domain boundaries, this special domain was formed to allow inverse mapping. The IN-ADDR.ARPA domain has four labels preceding it. These labels correspond to the 4 octets of an Internet address. All four octets must be specified even if an octets is zero. The Internet address `128.32.0.4` is located in the domain `4.0.32.128.IN-ADDR.ARPA`. This reversal of the address is awkward to read but allows for the natural grouping of hosts in a network.

5.4 Standard Resource Record Format

The records in the name server data files are called resource records. The Standard Resource Record Format (RR) is specified in RFC882 and RFC973. The following is a general description of these records:

```
{name}    {ttl} addr-classRecord Type Record Specific data
```

Resource records have a standard format shown above. The first field is always the name of the domain record and it must always start in column 1. For some RR's the name may be left blank; in that case it

takes on the name of the previous RR. The second field is an optional time to live field. This specifies how long this data will be stored in the data base. By leaving this field blank the default time to live is specified in the Start Of Authority resource record (see below). The third field is the address class; there are currently two classes: `IN` for internet addresses and `ANY` for all address classes. The fourth field states the type of the resource record. The fields after that are dependent on the type of the RR. Case is preserved in names and data fields when loaded into the name server. All comparisons and lookups in the name server data base are case insensitive.

The following characters have special meanings:

- `.` A free standing dot in the name field refers to the current domain.
- `@` A free standing @ in the name field denotes the current origin.
- `..` Two free standing dots represent the null domain name of the root when used in the name field.
- `\\X` Where X is any character other than a digit (0-9), quotes that character so that its special meaning does not apply. For example, "\e." can be used to place a dot character in a label.
- `\\DDD` Where each D is a digit, is the octet corresponding to the decimal number described by DDD. The resulting octet is assumed to be text and is not checked for special meaning.
- `()` Parentheses are used to group data that crosses a line. In effect, line terminations are not recognized within parentheses.
- `;` Semicolon starts a comment; the remainder of the line is ignored.
- `*` An asterisk signifies wildcarding.

Most resource records will have the current origin appended to names if they are not terminated by a ".". This is useful for appending the

current domain name to the data, such as machine names, but may cause problems where you do not want this to happen. A good rule of thumb is that, if the name is not in of the domain for which you are creating the data file, end the name with a ".".

5.4.1 $INCLUDE

An include line begins with `$INCLUDE`, starting in column 1, and is followed by a file name. This feature is particularly useful for separating different types of data into multiple files. An example would be:

```
$INCLUDE /usr/named/data/mailboxs
```

The line would be interpreted as a request to load the file `/usr/named/data/mailboxes`. The `$INCLUDE` command does not cause data to be loaded into a different zone or tree. This is simply a way to allow data for a given zone to be organized in separate files. For example, mailbox data might be kept separately from host data using this mechanism.

5.4.2 $ORIGIN

The origin is a way of changing the origin in a data file. The line starts in column 1, and is followed by a domain origin. This is useful for putting more then one domain in a data file.

5.4.3 SOA: start of authority

```
name    {ttl}   addr-class   SOA      Origin                  Person in charge
@               IN           SOA      ucbvax.Berkeley.Edu.kjd.ucbvax.Berkeley.Edu. (
                             1.1      ; Serial
                             3600     ; Refresh
                             300      ; Retry
                             3600000  ; Expire
                             3600 )   ; Minimum
```

The start of authority, `SOA`, record designates the start of a zone. The name is the name of the zone. Origin is the name of the host on which this data file resides. Person in charge is the mailing address for the person responsible for the name server. The serial number is the

version number of this data file, this number should be incremented whenever a change is made to the data. The name server cannot handle numbers over 9999 after the decimal point. The refresh indicates how often, in seconds, a secondary name servers is to check with the primary name server to see if an update is needed. The retry indicates how long, in seconds, a secondary server is to retry after a failure to check for a refresh. Expire is the upper limit, in seconds, that a secondary name server is to use the data before it expires for lack of getting a refresh. Minimum is the default number of seconds to be used for the time to live field on resource records. There should only be one SOA record per zone.

5.4.4 NS: Name server

```
{name}    {ttl}    addr-class    NS    Name servers name
                   IN            NS    ucbarpa.Berkeley.Edu.
```

The name server record, NS, lists a name server responsible for a given domain. The first name field lists the domain that is serviced by the listed name server. There should be one NS record for each Primary Master server for the domain.

5.4.5 A: address

```
{name}      {ttl}    addr-class    A    address
ucbarpa              IN            A    128.32.0.4
                     IN            A    10.0.0.78
```

The address record, A, lists the address for a given machine. The name field is the machine name and the address is the network address. There should be one A record for each address of the machine.

5.4.6 HINFO: host information

```
{name}    {ttl}    addr-class    HINFO          Hardware    OS
          ANY      HINFO         VAX-11/780     UNIX
```

`Host Information` resource record, `HINFO`, is for host specific data. This lists the hardware and operating system that are running at the listed host. It should be noted that only a single space separates the hardware info and the operating system info. If you want to include a space in the machine name you must quote the name. Host information is not specific to any address class, so `ANY` may be used for the address class. There should be one `HINFO` record for each host.

5.4.7 `WKS`: well-known services

{name}	*{ttl}*	*addr-class*	*WKS*	*address*	*protocol*	*list of services*
		IN	WKS	128.32.0.10	UDP	who route timed domain
		IN	WKS	128.32.0.10	TCP	echo telnet discard sunrpc sftp uucp-path systat daytime netstat qotd nntp link chargen ftp auth time whois mtp pop rje finger smtp supdup hostnames domain name server

The well-known services record, `WKS`, describes the well-known services supported by a particular protocol at a specified address. The list of services and port numbers come from the list of services specified in `/etc/services`. There should be only one `WKS` record per protocol per address.

5.4.8 `CNAME`: canonical name

aliases	*{ttl}*	*addr-class*	*CNAME*	*Canonical name*
ucbmonet		IN	CNAME	monet

The canonical name resource record, `CNAME`, specifies an alias for a canonical name. An alias should be the only record associated with the alias name; all other resource records should be associated with the canonical name and not with the alias. Any resource records that include a domain name as their value (for instance, NS or MX) should

list the canonical name, not the alias.

5.4.9 PTR: domain name pointer

name	*{ttl}*	*addr-class*	*PTR*	*real name*
7.0		IN	PTR	monet.Berkeley.Edu.

A domain name pointer record, PTR, allows special names to point to some other location in the domain. The above example of a PTR record is used in setting up reverse pointers for the special IN-ADDR.ARPA domain. This line is from the example hosts.rev file. PTR names should be unique to the zone.

5.4.10 MB: Mailbox

name	*{ttl}*	*addr-class*	*MB*	*Machine*
miriam		IN	MB	vineyd.DEC.COM.

MB is the mailbox record. This lists the machine where a user wants to receive mail. The name field is the users login; the machine field denotes the machine to which mail is to be delivered. Mail Box names should be unique to the zone.

5.4.11 MR: Mail rename name

Cname	*{ttl}*	*addr-class*	*MR*	*corresponding MB*
Postmistress		IN	MR	miriam

The main rename records, MR, can be used to list aliases for a user. The name field lists the alias for the name listed in the fourth field, which should have a corresponding MB record.

5.4.12 MINFO: Mailbox information

Cname	*{ttl}*	*addr-class*	*MINFO*	*requests*	*maintainer*
BIND		IN	MINFO	BIND-REQUEST	kjd.Berkeley.Edu.

The mail information record, MINFO, creates a mail group for a mailing list. This resource record is usually associated with a mail group `mail group`, but may be used with a `mail box` record. The `name` specifies the name of the mailbox. The `xrequests` field is where mail such as requests to be added to a mail group should be sent. The `maintainer` is a mailbox that should receive error messages. This is particularly appropriate for mailing lists when errors in members names should be reported to a person other than the sender.

5.4.13 MG: Mail group member

{mail group name}	*{ttl}*	*addr-class*	*MG*	*member name*
		IN	MG	Bloom

The mail group record, MG, lists members of a mail group.

An example for setting up a mailing list is as follows:

Bind	*IN*	*MINFO*	*Bind-Request*	*kjd.Berkeley.Edu.*
	IN	MG	Ralph.Berkeley.Edu.	
	IN	MG	Zhou.Berkeley.Edu.	
	IN	MG	Painter.Berkeley.Edu.	
	IN	MG	Riggle.Berkeley.Edu.	
	IN	MG	Terry.pa.Xerox.Com.	

5.4.14 MX: Mail exchanger

name	*{ttl}*	*addr-class*	*MX*	*preference value*	*mailer exchanger*
Munnari.OZ.AU.		IN	MX	0	Seismo.CSS.GOV.
*.IL.		IN	MX	0	RELAY.CS.NET.

Main exchanger records, MX, are used to specify a machine that knows how to deliver mail to a machine that is not directly connected to the network. In the first example, above, Seismo.CSS.GOV. is a mail gateway that knows how to deliver mail to Munnari.OZ.AU. but other machines on the network can not deliver mail directly to

Munnari. These two machines may have a private connection or use a different transport medium. The preference value is the order that a mailer should follow when there is more then one way to deliver mail to a single machine. See RFC974 for more detailed information.

Wildcard names containing the character * may be used for mail routing with MX records. There are likely to be servers on the network that simply state that any mail to a domain is to be routed through a relay. Second example, above, all mail to hosts in the domain IL is routed through `RELAY.CS.NET`. This is done by creating a wildcard resource record, which states that *.IL has an MX of `RELAY.CS.NET`.

5.5 Sample files

The following section contains sample files for the name server. This covers example boot files for the different types of servers and example domain data base files.

5.5.1 Boot file

5.5.1.1 Primary master server

```
;
; Boot file for Primary Master Name Server
;

; type          domain              source file or host
;
directory   /usr/local/domain
primary     Berkeley.Edu        ucbhosts
primary     32.128.in-addr.arpa ucbhosts.rev
primary     0.0.127.in-addr.arpa named.local
cache       .                   root.cache
```

5.5.2 Secondary master server

```
;
; Boot file for Primary Master Name Server
```

```
;

; type    domain          source file or host
;
directory /usr/local/domain
secondary Berkeley.Edu     128.32.0.4 128.32.0.10 ucbhosts.bak
secondary 32.128.in-addr.arpa 128.32.0.4 128.32.0.10 ucbhosts.rev.bak
primary   0.0.127.in-addr.arpa named.local
cache     .               root.cache
```

5.5.2.1 Caching-only server

```
;
; Boot file for Caching-only Name Server
;

; type      domain                source file or host
;
directory   /usr/local/domain
cache       .                     root.cache
primary     0.0.127.in-addr.arpa  /etc/named.local
```

5.5.3 Remote server

5.5.3.1 /etc/resolv.conf

```
domain Berkeley.Edu
name server 128.32.0.4
name server 128.32.0.10
```

5.5.4 root.cache

```
;
;
; Initial cache data for root domain servers.
;

.                       99999999    IN    NS    SRI-NIC.ARPA.
                        99999999    IN    NS    NS.NASA.GOV.
                        99999999    IN    NS    TERP.UMD.EDU.
                        99999999    IN    NS    A.ISI.EDU.
                        99999999    IN    NS    BRL-AOS.ARPA.
                        99999999    IN    NS    GUNTER-ADAM.ARPA.
                        99999999    IN    NS    C.NYSER.NET.

;  Prep the cache (hotwire the addresses).
SRI-NIC.ARPA.           99999999    IN    A     10.0.0.51
SRI-NIC.ARPA.           99999999    IN    A     26.0.0.73
NS.NASA.GOV.            99999999    IN    A     128.102.16.10
A.ISI.EDU.              99999999    IN    A     26.3.0.103
BRL-AOS.ARPA.           99999999    IN    A     128.20.1.2
BRL-AOS.ARPA.           99999999    IN    A     192.5.25.82
BRL-AOS.ARPA.           99999999    IN    A     192.5.22.82
GUNTER-ADAM.ARPA.       99999999    IN    A     26.1.0.13
C.NYSER.NET.            99999999    IN    A     128.213.5.17
TERP.UMD.EDU.           99999999    IN    A     10.1.0.17
```

5.5.5 named.local

```
@   IN   SOA   ucbvax.Berkeley.Edu. kjd.ucbvax.Berkeley.Edu. (
                1          ; Serial
```

```
                3600    ; Refresh
                300     ; Retry
                3600000 ; Expire
                3600 )  ; Minimum
    IN  NS      ucbvax.Berkeley.Edu.
1   IN  PTR     localhost.
```

5.5.6 hosts

```
;
;    @(#)ucb-hosts    1.1    (berkeley)    86/02/05
;

@                   IN        SOA         ucbvax.Berkeley.Edu. kjd.monet.Berkeley.Edu.
                                          1.1        ; Serial
                                          10800      ; Refresh
                                          1800       ; Retry
                                          3600000    ; Expire
                                          86400    ; Minimum
                    IN        NS          ucbarpa.Berkeley.Edu.
                    IN        NS          ucbvax.Berkeley.Edu.
localhost           IN        A           127.1
ucbarpa             IN        A           128.32.4
                    IN        A           10.0.0.78
                    ANY       HINFO       VAX-11/780 UNIX
arpa                IN        CNAME       ucbarpa
ernie               IN        A           128.32.6
                    ANY       HINFO       VAX-11/780 UNIX
ucbernie            IN        CNAME       ernie
monet               IN        A           128.32.7
                    IN        A           128.32.130.6
                    ANY       HINFO       VAX-11/750 UNIX
ucbmonet            IN        CNAME       monet
ucbvax              IN        A           10.2.0.78
                    IN        A           128.32.10
                    ANY       HINFO       VAX-11/750 UNIX
                    IN        WKS         128.32.0.10 UDP syslog route timed domain
                    IN        WKS         128.32.0.10 TCP ( echo telnet
                                          discard sunrpc sftp
                                          uucp-path systat daytime
                                          netstat qotd nntp
                                          link chargen ftp
                                          auth time whois mtp
                                          pop rje finger smtp
```

```
                                    supdup hostnames
                                    domain
                                    name server )
vax                 IN      CNAME   ucbvax
toybox              IN      A       128.32.131.119
                    ANY     HINFO   Pro350 RT11
toybox              IN      MX      0  monet.Berkeley.Edu
miriam              ANY     MB      vineyd.DEC.COM.
postmistress        ANY     MR      Miriam
Bind                ANY     MINFO   Bind-Request kjd.Berkeley.Edu.
                    ANY     MG      Ralph.Berkeley.Edu.
                    ANY     MG      Zhou.Berkeley.Edu.
                    ANY     MG      Painter.Berkeley.Edu.
                    ANY     MG      Riggle.Berkeley.Edu.
                    ANY     MG      Terry.pa.Xerox.Com.
```

5.5.7 host.rev

```
;
;    @(#)ucb-hosts.rev    1.1    (Berkeley)    86/02/05
;

@       IN   SOA   ucbvax.Berkeley.Edu.
                   kjd.monet.Berkeley.Edu.
                   1.1      ; Serial
                   10800     ; Refresh
                   1800      ; Retry
                   3600000   ; Expire
                   86400    ; Minimum
        IN   NS    ucbarpa.Berkeley.Edu.
        IN   NS    ucbvax.Berkeley.Edu.
4.0     IN   PTR   ucbarpa.Berkeley.Edu.
6.0     IN   PTR   ernie.Berkeley.Edu.
7.0     IN   PTR   monet.Berkeley.Edu.
10.0    IN   PTR   ucbvax.Berkeley.Edu.
6.130   IN   PTR   monet.Berkeley.Edu.
```

6. Domain management

This section contains information for starting, controlling and debugging named.

6.1 /etc/rc.local

The host name should be set to the full domain style name in /etc/rc.local using hostname (1). The following entry should be added to /etc/rc.local to start up named at system boot time:

```
if [ -f /etc/named ]; then
      /etc/named [options] & echo -n ' named' >/dev/console
fi
```

This usually directly follows the lines that start syslogd. **Do not** attempt to run named from inetd. This will continuously restart the name server and defeat the purpose of having a cache.

6.2 /etc/named.pid

When named is successfully started up it writes its process ID into the file /etc/named.pid. This is useful to programs that want to send

signals to named. The name of this file may be changed by defining PIDFILE to the new name when compiling named.

6.3 /etc/hosts

The gethostbyname () library call can detect if named is running. If it is determined that named is not running it will look in /etc/hosts to resolve an address. This option was added to allow ifconfg (8C) to configure the machines local interfaces and to enable a system manager to access the network while the system is in single user mode. It is advisable to put the local machines interface addresses and a couple of machine names and address in /etc/hosts so the system manager can rcp files from another machine when the system is in single user mode. The format of /etc/host has not changed. See hosts (5) for more information. Since the process of reading /etc/hosts is slow, it is not advised to use this option when the system is in multi user mode.

6.4 Signals

There are several signals that can be sent to the named process to have it do tasks without restarting the process.

6.4.1 Reload

SIGHUP

Causes named to read named.boot and reload the database. All previously cached data is lost. This is useful when you have made a change to a data file and you want named's internal database to reflect the change.

6.4.2 Debugging

When named is running incorrectly, look first in /usr/adm/messages and check for any messages logged by syslog. Next send it a signal to see what is happening.

SIGINT

Dumps the current data base and cache to /usr/ tmp/ named_dump . db This should give you an indication to whether the data base was loaded correctly. The name of the dump file may be changed by defining DUMPFILE to the new name when compiling named.

Note: The following two signals only work when `named` is built with `DEBUG` defined.

`SIGUSR1`
: Turns on debugging. Each following USR1 increments the debug level. The output goes to `/usr/tmp/named.run` The name of this debug file may be changed by defining `DEBUGFILE` to the new name before compiling `named`.

`SIGUSR2`
: Turns off debugging completely.

For more detailed debugging, define DEBUG when compiling the resolver routines into `/lib/libc.a`.

ACKNOWLEDGMENTS

Many thanks to the users at U.C. Berkeley for falling into many of the holes involved with integrating BIND into the system so that others would be spared the trauma. I would also like to extend gratitude to Jim McGinness and Digital Equipment Corporation for permitting me to spend most of my time on this project.

Ralph Campbell, Doug Kingston, Craig Partridge, Smoot Carl-Mitchell, Mike Muuss and everyone else on the DARPA Internet who has contributed to the development of BIND. To the members of the original BIND project, Douglas Terry, Mark Painter, David Riggle and Songnian Zhou.

Anne Hughes, Jim Bloom and Kirk McKusick and the many others who have reviewed this paper giving considerable advice.

This work was sponsored by the Defense Advanced Research Projects Agency (DoD), Arpa Order No. 4871 monitored by the Naval Electronics Systems Command under contract No. N00039-84-C-0089. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of the Defense Research Projects Agency, of the US Government, or of Digital Equipment Corporation.

REFERENCES

Birrell, A. D., Levin, R., Needham, R. M., and Schroeder, M.D., In *Comm. A.C.M. 25*, 4:260-274 April 1982. Su, Z. Postel, J., *Internet Request For Comment 819* Network Information Center, SRI International, Menlo Park, California. August 1982. Mockapetris, P., *Internet Request For Comment 973* Network Information Center, SRI International, Menlo Park, California. February 1986. Partridge, C., *Internet Request For Comment 974* Network Information Center, SRI International, Menlo Park, California. February 1986. Stahl, M., *Internet Request For Comment 1032* Network Information Center, SRI International, Menlo Park, California. November 1987. Lottor, M., *Internet Request For Comment 1033* Network Information Center, SRI International, Menlo Park, California. November 1987. Mockapetris, P., *Internet Request For Comment 1034* Network Information Center, SRI International, Menlo Park, California. November 1987. Mockapetris, P., *Internet Request For Comment 1035* Network Information Center, SRI International, Menlo Park, California. November 1987. Terry, D. B., Painter, M., Riggle, D. W., and Zhou, S., *The Berkeley Internet Name Domain Server*. Proceedings USENIX Summer Conference, Salt Lake City, Utah. June 1984, pages 23-31. Zhou, S., *The Design and Implementation of the Berkeley Internet Name Domain (BIND) Servers*. UCB/CSD 84/177. University of California, Berkeley, Computer Science Division. May 1984.

Appendix C
Additional Reading

Contents

1. Related manual page entries 1
   1.1 User commands 1
   1.2 Administrative commands 2
   1.3 System calls 3
   1.4 Subroutines 4
   1.5 File formats 5
   1.6 Protocols 5
2. Related RFCs 5
3. Berkeley documentation 7

Appendix C

Additional Reading

This appendix lists additional reading material that you might find helpful. The first section lists relevant man pages in the area of user commands, administrative commands, subroutines, file formats, and protocols. The next sections list related requests for comments (RFCs) and related documentation from the University of California at Berkeley.

1. Related manual page entries

The commands, system calls, subroutines, and system files described on the manual pages listed here are either directly related to the networking or NFS/Yellow Pages implementation or are NFS versions of standard UNIX programs. All of the manual page entries listed here may be viewed on the screen by entering

`man` *command-name*

The printed copies of these pages are in *A/UX Command Reference* (Section 1), *A/UX System Administrator's Reference* (Section 1M), and *A/UX Programmer's Reference* (Sections 2, 3, 4, and 5).

1.1 User commands

`domainname`(1)	Set or display name of current domain system.
`ftp`(1N)	File transfer program.
`hostid`(1N)	Set or print identifier of current host system.
`hostname`(1N)	Set or print name of current host system.
`netstat`(1N)	Show network status.
`rcp`(1N)	Remote file copy.
`remsh`(1N)	Remote shell.
`rlogin`(1N)	Remote login.
`ruptime`(1N)	Show host status of local machines.
`rusers`(1N)	Who's logged in on local machines (RPC version).

`rwho`(1N)	Who's logged in on local machines.
`talk`(1N)	Talk to another user.
`telnet`(1N)	User interface to the TELNET protocol.
`ypcat`(1)	Print values in a Yellow Pages database.
`ypmatch`(1)	Print values of one or more keys from a Yellow Pages map.
`yppasswd`(1)	Change login password in Yellow Pages.
`ypwhich`(1)	Which host is the Yellow Pages server or map master?

1.2 Administrative commands

`ftpd`(1M)	DARPA Internet File Transfer Protocol server.
`ifconfig`(1M)	Configure network interface parameters.
`makedbm`(1M)	Make a Yellow Pages `dbm` file (see also `ypmake`(1M)).
`mountd`(1M)	NFS mount request server if listed in `/etc/servers`.
`nfsd`(1M)	NFS daemons.
`nfsstat`(1M)	Print NFS statistics.
`ping`(1M)	Send ICMP `ECHO_REQUEST` packets to network hosts.
`portmap`(1M)	DARPA port to RPC program number mapper.
`remshd`(1M)	Remote shell server.
`rexecd`(1M)	Remote execution server.
`rdump`(1M)	Back up to a remote device
`rlogind`(1M)	Remote login server.
`route`(1M)	Manually manipulate routing tables.
`rpcinfo`(1M)	Print RPC information.
`routed`(1M)	Network routing daemon.
`rrestore`(1M)	Restore from remote backup medium.
`rstatd`(1M)	Kernel statistics server.
`rusersd`(1M)	`rusers` server.
`rwall`(1M)	Write to all users over a network.
`rwalld`(1M)	Network `wall` server.
`rwhod`(1M)	System status server.

showmount(1M)	Show all remote mounts.
spray(1M)	spray packets.
sprayd(1M)	spray server.
telnetd(1M)	DARPA TELNET protocol server.
tftpd(1M)	DARPA Trivial File Transfer Protocol server.
trpt(1M)	Transliterate protocol trace.
ypinit(1M)	Build and install Yellow Pages database.
ypmake(1M)	Rebuild Yellow Pages database by using /etc/yp/Makefile.
yppasswdd(1M)	Server for modifying Yellow Pages password file.
yppoll(1M)	Print which version of a Yellow Pages map is at a Yellow Pages server host.
yppush(1M)	Force propagation of a changed Yellow Pages map.
ypserv(1M)	Yellow Pages server and binder processes.
ypset(1M)	Point ypbind at a particular server.
ypxfr(1M)	Transfer a Yellow Pages map from a Yellow Pages server to here.

1.3 System calls

accept(2N)	Accept connection on a socket.
bind(2N)	Bind name to a socket.
connect(2N)	Initiate connection on a socket.
fcntl(2)	File control.
fsmount(2)	Mount NFS file system.
getdirentries(2)	Get directory entries in a file system-independent format.
getdomainname(2)	Get current domain name.
getdtablesize(2)	Get descriptor table size.
gethostid(2N)	Get/set unique ID of current host.
gethostname(2N)	Get/set name of current host.
getpeername(2N)	Get name of connected peer.
getsockname(2N)	Get socket name.
getsockopt(2N)	Get and set options on sockets.
listen(2N)	Listen for connections on sockets.
locking(2)	Provide exclusive file regions for reading or writing.

nfssvc(2)	NFS daemons.
readlink(2)	Read value of a symbolic link.
recv(2N)	Receive message from a socket.
rename(2)	Change name of a file.
select(2N)	Synchronous I/O multiplexing.
setregid(2)	Set real and effective group IDs.
setreuid(2)	Set real and effective user IDs.
send(2N)	Send message from a socket.
shutdown(2N)	Shut down part of a full-duplex connection.
socket(2N)	Create endpoint for communication.
stat(2)	Get file status.
statfs(2)	Get file system statistics.
symlink(2)	Make symbolic link to a file.
truncate(2)	Truncate file to a specified length.
uvar(2)	Return system-specific configuration information.
wait3(2)	Wait for child process to stop or terminate.

1.4 Subroutines

bcopy(3N)	Bit and byte string operations.
byteorder(3N)	Convert values between host and network byte order.
dbm(3X)	Database subroutines.
directory(3)	Directory operations.
dup2(3N)	Duplicate descriptor.
getgrent(3C)	Obtain group file entry from a group file.
gethostbyaddr(3N)	Get network host entry.
getmntent(3)	Get file system descriptor file entry.
getnetgrent(3N)	Get network group entry.
getprotoent(3N)	Get protocol entry.
getpwent(3C)	Get password file entry.
getservent(3N)	Get service entry.
inet_addr(3N)	Internet address manipulation routines.
initgroups(3)	initialize group access list.
insque(3N)	Insert/remove element from a queue.
killpg(3N)	Send signal to a process group.
lockf(3C)	Record locking on files.

ypclnt(3N)	Yellow Pages libraries.
rcmd(3N)	Routines for returning a stream to a remote command.
rexec(3N)	Return stream to a remote command.
setuid(3)	Set user and group IDs.

1.5 File formats

dir(4)	Format of System V directories.
exports(4)	List NFS file systems being exported and who has permission to mount them.
fs(4)	Format of a System V system volume.
fstab(4)	Static information about file systems.
group(4)	Group file.
hosts(4N)	Host name database.
hosts.equiv(4)	List of trusted hosts.
inode(4)	Format of a System V inode.
mtab(4)	Mounted file system table.
netgroup(4)	List of network-wide groups of machines and users.
networks(4N)	Network name database.
passwd(4)	Password file.
protocols(4N)	Protocol name database.
rmtab(4)	Remotely mounted file system table.
servers(4N)	Consulted by network super-daemon inetd for list of daemons to start up.
services(4N)	Service name database.
ypfiles(4)	The Yellow Pages database and directory structure.

1.6 Protocols

arp(5P)	Address Resolution Protocol.
inet(5F)	Internet protocol family.
ip(5P)	Internet Protocol.
tcp(5P)	Internet Transmission Control Protocol.
udp(5P)	Internet User Datagram Protocol.

2. Related RFCs

The following requests for comments (RFCs) provide additional information about the networking implementation. You can obtain

these documents by writing or telephoning

Network Information Center
SRI International
Menlo Park, CA 94025
(800) 235-3155

TCP (Internet Transmission Control Protocol) RFC 793, September 1981.

IP (Internet Protocol) Postel, J., *Internet Protocol*, RFC 791, USC/Information Sciences Institute, September 1981.

ICMP
Postel, J., *Internet Control Message Protocol*, RFC 792, USC/Information Sciences Institute, September 1981.

UDP (Internet User Datagram Protocol) RFC 768, August 1980.

FTP (File Transfer Protocol) RFC 959, October 1985.

ARP (Address Resolution Protocol) RFC 826, November 1982.

TELNET
RFC 854, May 1983.

Subnets
Mogul, J., and J. Postel, *Internet Standard Subnetting Procedure*, RFC 950, Stanford University, August 1985.

Mogul, J., *Internet Subnets*, RFC 917, Stanford University, October 1984.

GADS, *Towards an Internet Standard Scheme for Subnetting*, RFC 940, Network Information Center, SRI International, April 1985.

Subnets and broadcasting
Mogul, J., *Broadcasting Internet Datagrams*, RFC 919, Stanford University, October 1984.

Mogul, J., *Broadcasting Internet Datagrams in the Presence of Subnets*, December 1987.

Internet domains
Lottor, M.K., *Domain Administrators Operations Guide*, SRI International, September 1987.

Mockapetris, P., *Domain System Changes and Observations*, RFC 973, January 1986.

Partridge, C., *Mail Routing and the Domain System*, RFC 974, January 1986.

Mockapetris, P., *Domain Names — Implementation Specification*, RFC 883, November 1983.

Mockapetris, P., *Domains Names — Concepts and Facilities*, RFC 882, November 1983.

Network mail

Su, Z., and J. Postel, *The Domain Naming Convention for Internet User Applications*, RFC 819, August 1982.

Crocker, D., *Standard for the Format of ARPA Internet Text Messages*, RFC 822, August 1982.

Postel, J., *Simple Mail Transfer Protocol*, RFC 821, August 1982.

Network numbers

Reynolds, J., and J. Postel, *Assigned Numbers*, RFC 960, USC/Information Sciences Institute, December 1985.

3. Berkeley documentation

The following documentation provides additional information about the network mail and Internet domains. To purchase these documents, you must be a current USENIX member:

USENIX Association
P.O. Box 2299
Berkeley, California 94710

`sendmail`

Allman, E., *Mail Systems and Addressing in 4.2BSD*, University of California, Berkeley.

Allman, E., *Sendmail — An Internetwork Mail Router*, University of California, Berkeley.

THE APPLE PUBLISHING SYSTEM

This Apple manual was written, edited, and composed on a desktop publishing system using Apple Macintosh® computers and Microsoft® Word. Proof pages were created on the Apple LaserWriter® Plus. Final pages were created on the Varityper® VT600™. POSTSCRIPT®, the LaserWriter page-description language, was developed by Adobe Systems Incorporated.

Text type is ITC Garamond® (a downloadable font distributed by Adobe Systems). Display type is ITC Avant Garde Gothic®. Bullets are ITC Zapf Dingbats®. Some elements, such as program listings, are set in Apple Courier, a fixed-width font.

2/21/88